2-(mix)

DESIGN OF A DOCUMENT RETRIEVAL SYSTEM USING PATTERN RECOGNITION

AND MATHEMATICAL PROGRAMMING TECHNIQUES

by

Steven R. Borbash, Jr.

B.S., Ohio University, 1958

M.S. in E.S., University of Toledo, 1966

Submitted to the Graduate Faculty

of the School of Engineering

in partial fulfillment of

the requirements for the degree of

Doctor

of

Philosophy

University of Pittsburgh

1970

" ABSTRACT FOR STAR. - AUTHOR SUGGESTS USE OF TITLE FOR BIBLIOGRAPHIC CITATION. X-52747

DESIGN OF A DOCUMENT RETRIEVAL SYSTEM USING PATTERN RECOGNITION AND MATHEMATICAL PROGRAMMING TECHNIQUES

by Steven R. Borbash, Jr.

A training set (TS) of document records with assigned utilities and a utility threshold defining document relevance are provided by a user. The TS is processed to give Boolean combinations of index terms for searching a document file. A linear utility prediction function (LUPF) is fitted to the TS documents using selected index terms. The LUPF is thresholded and the resulting pseudo-Boolean inequality is solved, giving term combinations for retrieving relevant documents. Algorithms are presented and testing is discussed.

NASA Index Terms

Category 19

Algorithm
*Boolean functions
Discrimination
Factorial design
Inequalities
*Information retrieval
Information theory
Linear programming
*Operations research
*Pattern recognition
Searching

*Main terms

FOREWORD

The author wishes to express his gratitude to the National Aeronautics and Space Administration, Lewis Research Center, Cleveland, Ohio for their financial and other support of this work.

The author is indebted to the Knowledge Availability Systems (KAS) Center and the Department of Industrial Engineering at the University of Pittsburgh for providing him with the opportunity to become familiar with the details and problems associated with operating a document retrieval system. The author would also like to express his appreciation for the fine support programming and other help contributed by Mr. Paul Peters and other members of the KAS Center staff during the testing phase of this work.

Finally, the author wishes to acknowledge the general encouragement and many evenings of typing contributed by his wife.

Oberlin, Ohio
July, 1970

ABSTRACT

Signature *Yuan Kem Wang*

DESIGN OF A DOCUMENT RETRIEVAL SYSTEM USING PATTERN RECOGNITION AND MATHEMATICAL PROGRAMMING TECHNIQUES

Steven R. Borbash, Jr., Ph.D.

University of Pittsburgh, 1970

A pattern recognition (PR) model of the document retrieval process is introduced. This model processes a training set (TS) of documents to derive file searching instructions. A file of indexed documents and a subsystem to implement search instructions is assumed to be available. Documents are represented as binary vectors of index terms. Two mutually exclusive categories of documents exist, A (relevant) or B (non-relevant). Each document in the TS is assigned a utility u on an arbitrary scale by a user. All documents in the TS with $u \geq \tau$ (a user specified threshold) are relevant.

The system 'learns' from the TS to predict document utility as a linear function of the index terms and hence to recognize relevant documents. The TS is processed by feature extraction followed by estimation of parameters in the linear utility prediction function

(LUPF). Feature extraction discards all but those index terms judged 'best' using an information theoretic estimate. The LUPF parameters are those which give a 'best' approximation (in the L_1 norm sense) to the utilities of the TS documents as a function of the extracted index terms. This approximation problem is solved as a linear programming problem.

After the LUPF has been estimated, relevant documents can be identified by applying the LUPF and the threshold τ sequentially to all document vectors in the file. This is a 'weighted term' search. Equivalent Boolean search instructions (called a Boolean retrieval strategy or BRS) can be derived by solving the linear pseudo-Boolean inequality (LPBI) formed by the LUPF and the threshold. The solution to this LPBI is a group of index term combinations (solution families). All documents having index term combinations which match any one of the solution families will be relevant. Each solution family may be regarded as a 'matching template' for classifying pattern vectors. This analytical derivation of the BRS shows the relation between 'weighted term' and 'Boolean' searches. Other methods of BRS construction are subjective. An algorithm is given for solving the LPBI which explores a binary tree using a branch and exclude technique.

The PR model was tested on the NASA document file using a designed factorial experiment. Human analysts and the PR system both produced BRS's from the same training sets. The effectiveness of

searches done with these BRS's were compared. Human analysts were approximately twice as effective as the automatic PR system. The analysts supplement their TS's with extra information not available to the PR system. Suggestions for improving the PR system are offered.

DESCRIPTORS

Analysis of variance	Algorithms
Boolean functions	Discrimination
Inequalities	Information retrieval
Information retrieval effectiveness	Information theory
Linear programming	Operations research
Pattern recognition	Patterns
Search structuring	Trees (mathematics)

TABLE OF CONTENTS

FOREWORD ii

ABSTRACT iii

LIST OF FIGURES xvi

LIST OF TABLES xix

GLOSSARY xx

1.0 INTRODUCTION 1

1.1 Summary 1

1.11 Objective 1

1.12 Motivation 2

1.13 Relationship to the Work of Others 3

1.14 Methods Used 4

1.15 Testing and Results 6

1.16 Conclusions 6

1.2 Structure and Assumptions of the Model 8

1.21 Document Representation and File Structure . . . 8

1.22 Fundamental Assumptions 10

1.221 File Existence 10

1.222 Document Utility 10

1.223 Document Relevance 10

1.224 System Objective 10

1.225 Source of Information for Utility Prediction 10

1.226 Dimensionality Reduction 10

1.227 Linear Utility Function 11

1.228 Estimation of Parameters in the Linear Utility Function 11

1.3 Limitations 11

1.4 Organization of this Dissertation 12

1.5 Contributions 15

1.51 Models 15

1.52 Methods 16

1.53 Data 16

2.0 PATTERN RECOGNITION SYSTEMS 17

2.1 Introduction 17

2.2 Pre-Processing 18

2.21 Representation of the Pattern as a Vector 18

2.22 Feature Extraction 18

2.3 Decision Function Specification 20

2.31 Selecting the Form of Decision Function 21

2.32 Estimating the Parameters of the Decision Function 22

2.321 The Associated Approximation Problem . . . 22

2.322 Choosing the Criterion of Best Approximation 23

2.33 Current Methods of Forming Decision Functions . . 25

2.34 An Example Problem Illustrating Decision Function Determination 26

2.35 Relationship of the Decision Function to Curve-Fitting Problems 29

2.4 Template Matching Operations 30

2.41 Introduction 30

2.42 The Pseudo-Boolean Inequality 31

2.43 An Example of Classification by Template Matching 34

2.5 Summary 35

3.0 MODELING THE DOCUMENT RETRIEVAL PROCESS AS A PATTERN RECOGNITION SYSTEM 36

3.1 The Document Retrieval System 36

3.11 General 36

3.12 Representation of Documents in the File 37

3.13 Specification of File Search Instructions 37

3.14 Satisfying User Needs 38

3.15 Problem Areas 38

3.16 A Model of the Retrieval Process 40

3.2 An Associated Pattern Recognition System 40

3.21 Characterization of Documents as Pattern Vectors 40

3.22 Definition of Two Categories 41

3.23 The Configuration of the System 41

3.231 Training Set Formation 41

3.232 Feature Extraction 42

3.233 Decision Function Formation 42

3.3 Implementing the Decision Function 44

3.31 Direct Method 44

3.32 Indirect Method 45

3.33 Relation of Decision Function Implementation to Retrieval System File Structure 46

3.34 Example Showing System Operation 49

4.0 AN INFORMATION THEORETIC MEASURE FOR RANKING AND SELECTING INDEX TERMS 53

4.1 Introduction 53

4.2 The Decision Theory Model 54

4.21 Decision Problems with Experimentation 55

4.211 Bayes Rule 56

4.3 Selection of Experiments 58

4.31 Decision Theory Approach When the Utilities Are Known 58

4.32 Inadequacy of the Decision Theory Model when the Utilities Are Not Known 59

4.4 Results from Information Theory 60

4.41 Definition of Entropy 60

4.42 Definitions of Event Sets and Probabilities . . 62

4.43 Entropy of the Distributions 65

4.44 Useful Relationships between Entropies of Distributions 66

4.5 Interpretation of Information Theoretic Results . . . 67

4.51 Bayesian Interpretation 67

4.52 Communication Theory Interpretation 69

4.53 Computation of an Information Statistic $\hat{R}$ . . . 70

4.54 Statistical Distribution of the Information Statistic 73

4.55 Example Problem 74

5.0 SOLVING THE DISCRETE LINEAR APPROXIMATION PROBLEM IN THE L_1 NORM 76

5.1 Introduction 76

5.2 Formulating the Discrete L_1 Problem as a Linear Programming Problem 78

5.3 Solving the L_1 Problem 82

5.4 Example Problems 83

5.5 The Effects of Alternate Optima 87

5.6 Secondary Feature Extraction 90

5.7 More Efficient Algorithms 90

6.0 DETERMINATION OF THE OPTIMAL BRS 92

6.1 Scope and Organization 92

6.2 The LPBI Arising from the Document LUPF 93

6.3 Properties of the LPBI and Its Solutions 95

6.31 General Form of the LPBI 95

6.32 Canonical Form of the LPBI 97

6.321 Transformation of Parameters of the LPBI from the General Form to the Canonical Form 97

6.322 Transformation of Solutions of the LPBI from the Canonical Form to the General Form 99

6.323 Some Proofs of Results Related to the Transformations 101

6.33 Families of Solutions 107

6.34 The Relationship between Binary Trees and Solutions of a LPBI 108

6.341 Isomorphism of Tree Paths to Possible Solutions 108

6.342 Isomorphism of Tree Nodes to Partial Path Records and Partial Inequalities . 111

6.35 Solutions of the Canonical Form 112

6.351 Basic Solutions 112

6.352 Canonical Solution Families 115

6.36 Summary of Solution Procedure for the LPBI . . . 117

6.4 Determining Basic Solutions of the LPBI by Searching the Binary Solution Tree 117

6.41 Preview of the Tree Pruning Algorithm (TPA) . . . 117

6.5 Solution Construction and Node Visits 119

6.51 Constructing Complete BSP's from Partial BSP's . 119

6.52 Node Visits Summarized in Terms of PIN Parameters 121

6.6 Scheduling Node Visits in the Binary Solution Tree . . 121

6.61 A Simple TPA Example Problem 124

6.62 The Pre-Order Tree Traversal Algorithm (TTA) . . 126

6.63 Modifying the TTA to Permit Tree-Pruning 131

6.631 Elimination of the Pre-Determined Link Table 133

6.632 Storage Allocation Modifications 133

6.64 Maintaining the Dynamic PPR and PIN Records . . 135

6.641 Maintenance of the PPR 135

6.642 Maintenance of the PIN Coefficients . . . 140

6.7 The Tree-Pruning Algorithm (TPA) 142

6.71 Detailed TPA Description 142

6.72 Example Problems 145

6.721 A Detailed Example of the TPA 145

6.722 Solving the General Form Inequality . . . 146

6.73 Miscellaneous 153

6.74 The Use of Solution Families in a Document Retrieval System 156

6.75 Computer Implementation of the TPA 157

6.76 Computational Experience with the TPA 159

7.0 EXPERIMENT DESIGN AND PRESENTATION OF DATA 165

7.1 Test Objectives 165

7.2 Measuring Search Effectiveness 166

7.21 Recall and Precision 166

7.22 The Information Statistic as a Measure of Search Effectiveness 167

7.23 Other Applications of the Information Statistic. . 169

7.24 Summary 171

7.3 Experiment Design 172

7.31 General 172

7.32 Selection and Preparation of Test Questions . . . 172

7.33 Classification of Variables in the Problem 175

7.331 Independent Variables Controlled as Part of the Problem 175

7.332 Variables Held Constant as Boundary Conditions on the Problem 176

7.333 Uncontrolled Factors Contributing to the Error Variance 176

7.34 Factors and Variables Not Considered in the Experiment 176

7.35 The Model Equation and Expected Mean Squares Table 177

7.36 Choice of Sample Size 178

7.37 A Sub-Experiment to Determine the Effect of Analysts within Method 1 181

7.4 Presentation of the Experimental Data 182

7.41 Factorial Experiment Response Data 182

7.42 Latin Square Experiment Response Data 184

7.43 Predicted Document Utilities vs. Known Document Utilities 185

7.44 Values of $\hat{R}$ for Index Terms in the Training Sets 188

7.45 BRS Descriptions 188

8.0 EXPERIMENTAL DATA ANALYSIS 192

8.1 Analysis of Variance for the Factorial Experiments . . . 192

8.11 Dependence of the NIS on Methods 192

8.12 Dependence of Precision on Methods 195

8.13 Dependence of Recall on Training Set Size 196

8.14 Lack of an Effect Due to the Number of BRS Index Terms 197

8.15 Summary of the Factorial Experiment 198

8.2 Analysis of Variance for the Latin Square Sub-Experiment 199

8.3 Extraction of Best Index Terms 199

8.31 Distribution of $\hat{R}$ 199

8.32 Differences in Index Term Selection between Methods 200

8.321 Differences in $\hat{R}$ 200

8.322 Differences in Frequencies of Term Occurrence 201

8.33 The Sampling Problem 201

8.4 Analysis of the BRS 203

8.41 Dependence of BRS Solution Family Size on Methods 203

8.42 Effects of BRS Family Size on Retrieval System Operation 205

8.5 Predicted Utilities of Relevant Documents for M_2 . . . 206

8.51 Factors Affecting the Recall of the M_2 System 206

8.52 Effects of Increasing the Vocabulary Size 207

8.6 Summary of the Data Analysis 208

8.7 Conclusions 210

9.0 SUGGESTIONS FOR FURTHER RESEARCH 212

9.1 General 212

9.2 Modifications to the Information Statistic for Selecting Index Terms 213

9.21 Incorporating Information about Frequency of Term Occurrence 213

9.22 Utilizing More Refined Document Utility Measurements 213

9.3 Applying the Feature Selection Process to Different Types of Features 215

9.31 Higher Order Features 215

9.32 Selection of Features for Training Set Coverage . 216

9.33 Major and Minor Index Terms in the NASA System. . 217

9.4 Modifications to the BRS Structure 217

9.41 Avoiding Solution Families with S = 1 217

9.42 Solving the Nonlinear Pseudo-Boolean Inequality . 218

9.5 Derivation of the LUPF 219

9.51 Choice of Norm 220

9.52 Selection among Alternate Optimal Solutions . . . 220

9.53 Choice of Independent Variables 220

9.54 Choice of Dependent Variables 221

9.55 Linear Programming Problems with Unequal Slack Costs 221

9.551 Forced Fitting to the Relevant Documents 222

9.552 Application to Iterative Retrieval 222

9.56 Improved Algorithms 222

9.6 Experimental Investigation of Iterative Retrieval . . . 223

APPENDIX A - AN EXAMPLE PROBLEM 224

A.1 Input Data 224

A.2 Processing of Index Terms 225

A.3 The Document-Term Matrix and Computation of $\hat{R}$ 226

A.4 Solving the L_1 Norm Approximation Problem 226

A.5 Solving the LPBI 228

A.6 Miscellaneous Results 230

BIBLIOGRAPHY 256

LIST OF FIGURES

FIGURE	TITLE	PAGE
1-1	Block Diagram of a Document Retrieval System Using Pattern Recognition Techniques	7
2-1	Example Showing Patterns Represented as Vectors and Illustrating Feature Extraction	19
2-2	Example Showing Linear Decision Function Parameter Estimation	27
3-1	Equivalence of a Subjective BRS to a Union of Solution Families	47
3-2	Sample Problem Illustrating Derivation of the Decision Function and BRS	51
3-3	Predicted Utilities for Combinations of Index Terms. .	52
4-1	Entropy Plot of a Simple Binary Distribution as a Function of One Probability	63
4-2	Examples Illustrating Computation of an Information Statistic for Estimating Information about Document Relevance Conveyed by Index Terms	75
5-1	Sample L_1 Problem-Formulation (5-1)	84
5-2	Sample L_1 Problem - Formulation (5-1) Showing Alternate Optimal Tableau	85
5-3	Sample L_1 Problem - Formulation (5-2)	86
5-4	Effects of Alternate Optimal Solutions on Predicted Utilities	89
6-1	Flow Chart Showing Transformations Involved in the Solution of a Linear Pseudo-Boolean Inequality	105
6-2	Solution Tree and Associated Data for a Simple Inequality	110

FIGURE	TITLE	PAGE
6-3	Binary Solution Tree Associated with an LPBI in General Form	113
6-4	Binary Solution Tree Associated with an LPBI in Canonical Form	114
6-5	Solution Decision Table	122
6-6	Flow Chart Showing the Modified Solution Decision Table for Classifying Partial Inequalities	123
6-7	Binary Tree with Associated Link Diagram and Link Table	128
6-8	Flow Chart Showing an Algorithm for Pre-Order Traversal of Binary Trees	130
6-9	Example Problem Illustrating the Tree Pre-Order Traversal Algorithm	132
6-10	Flow Chart of the Tree Traversal Algorithm after Modification to Permit Generation of a Dynamic Link Table	136
6-11	Details of PER and PIN Dynamic Modifications	139
6-12	Flow Chart of the Tree Pruning Algorithm for Solving Pseudo-Boolean Inequalities in Canonical Form	141
6-13	Example Problem Showing Details of the Tree Pruning Algorithm for the Inequality $3x_1 + 2x_2 + x_3 \geq 4$	147
6-14	Continuation of Example Problem Showing Details of the Tree Pruning Algorithm	148
6-15	Example Problem Showing Transformations Involved in the Solution of a Linear Pseudo-Boolean Inequality	149
6-16	Example Problem Showing Enumeration of Solution Vectors Before and After Transformation	154
7-1	Recall and Precision vs. the Normalized Information Statistic (NIS) for N = 5100 and $n_{11} = 2$ (Relevant retrieved).	170

FIGURE	TITLE	PAGE
A-1 to A-4	Input Data for Sample Problem (Index Terms, Utilities, Document Numbers)	235 to 238
A-5	Document Data Summary for Sample Problem	239
A-6 to A-8	Alphabetical Listing of Index Terms in Sample Problem Training Set	240 to 242
A-9 to A-11	Information Statistic Sort of Index Terms in Sample Problem Training Set	243 to 245
A-12 to A-14	Document-Term Matrix for Sample Problem	246 to 248
A-15 to A-17	LP Structural Matrix for Sample Problem	249 to 251
A-18	LP Solution Summary	252
A-19	Computation of Residuals for Fitted Model	253
A-20	Initial and Canonical Form Coefficients for the Pseudo-Boolean Inequality	254
A-21	Solutions to the Pseudo-Boolean Inequality	255

LIST OF TABLES

TABLE	TITLE	PAGE
6-1	Data Illustrating Computational Experience with the TPA	160
6-2	Smoothed and Extrapolated Estimates of TPA Performance	162
7-1	Response Data from Factorial Experiment	183
7-2	Response Data for Latin Square Design within Method 1	184
7-3	Comparison of Predicted Document Utility with Actual Document Utility	187
7-4	Distribution of $\hat{R} = H(X) - H(X/Y)$ for Index Terms Appearing in the Training Set	188
7-5	Distribution of Solution Families Having SIZE S	191
8-1	Analysis of Variance Tables for Factorial Experiments	193
8-2	Analysis of Variance Table for Latin Square Experiment	199
8-3	Comparison of BRS Solution Family Sizes for M_1 and M_2	204

GLOSSARY

ITEM		PAGE
BRS	- Boolean Retrieval Strategy	1,37,46
	Basic Solution	112
BSP	- Basic Solution Path	112,115
	Canonical Inequality	97
	Canonical Solution Family	115
DRS	- Document Retrieval System	1,36
ISF	- Inversely Structured File	5,9,48
LP	- Linear Programming	24,78
LPBI	- Linear Pseudo-Boolean Inequality	5,31,95
LUPF	- Linear Utility Prediction Function	5,49,93
NIS	- Normalized Information Statistic	169,179
PIN	- Partial Inequality	111,120,140
PPR	- Partial Path Record	111,120,135
SSF	- Sequentially Structured File	5,9,48
TPA	- Tree Pruning Algorithm	13,92,118,124,134
TTA	- Tree Traversal Algorithm	124,126

1.0 INTRODUCTION

1.1 Summary

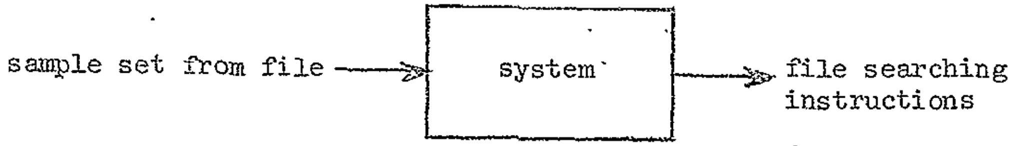
1.11 Objective

This dissertation presents details of the design and testing of a document retrieval system (DRS) using the NASA Scientific and Technical Information System[(1,2,3,4)*]. The analytical model used for the DRS treats the system as a pattern recognizer. The objective of the system is to automatically develop a set of Boolean file searching instructions from a sample of relevant and non-relevant documents.

A computerized file of document numbers and and associated index terms is assumed to be available. The system presented here receives as input a sample set of documents from this file. Each of the documents in the set has been assigned a personal utility by a user. In addition to the sample set, the user has specified a utility threshold τ, which defines two categories, relevant and not relevant.

The system output is a set of searching instructions for retrieving all other documents from the file which are predicted to be relevant, based on the examples provided in the sample set. The searching instructions are presented as Boolean combinations of index terms which are collectively known as a Boolean retrieval strategy (BRS). The system is shown on the next page.

*Parenthetical references placed superior to the line of text refer to the bibliography.



1.12 Motivation

A DRS which functions as described above provides a new method for a user to interact with a computerized file. This method eliminates some pressing practical problems. In addition, it provides a new analytical framework for studying the retrieval process.

There are practical problems associated with the present method of communication between the human user and the computerized file. The NASA system currently accepts file searching instructions in the form of a <u>subjectively</u> derived BRS submitted by a user. All documents which match this subjective BRS are then retrieved for the user.

To form a BRS the user first selects a small subset of index terms. Next the user specifies Boolean combinations of these terms which he feels are meaningful. As an aid to index term selection and combination, the user may consult a thesaurus and/or consider index term usage statistics. The subjective determination of a BRS in this manner is very difficult and fatiguing, and results are often unsatisfactory. New methods are needed which help the user select and combine terms..

The DRS presented here provides this type of aid to a user. A <u>training</u> or example set of documents is presented to the system. The DRS attempts to 'learn' how to discriminate between relevant and non-relevant documents by using this set. Thus the DRS becomes an intellectual tool of the user and acts as his 'agent' to derive a BRS. This system allows the user to concentrate his efforts on making value judgments of documents in the training set. It relieves him of the combinatorial problems of BRS formation.

Analytically, the model used here allows pattern recognition and mathematical programming techniques developed for pattern recognition systems to be applied directly to the document retrieval problem. In addition to supplying numerical techniques, the model suggests many extensions for further study.

1.13 Relationship to the Work of Others

The DRS model developed here fills an important gap in the literature. This results from concentrating only on deriving the BRS from the training set. Both automatic index term extraction and the techniques of carrying out search requests have been excluded from consideration here. A file of indexed documents is assumed to exist, along with a system for carrying out search instructions.

In other DRS's, automatic index term extraction from full English text has occupied a large portion of the analytical effort[5,6]. Still other researchers have been concerned mainly with the file structure and/or the mechanics of carrying out search requests[7,8]. Generally a specified set of search instructions is regarded as the input or query to their systems.

In the system here, the BRS is developed analytically from the training set by first deriving a set of index term weights and then developing the BRS from these. Others have used weighted term systems to carry out file searches. The index term weights are quite often assigned subjectively[9,10] and occasionally by analytical methods[11,12]. The analytical method used here to derive term weights is new, and depends upon user-assigned document utilities.

An important new result here is that the BRS is simply an alternate way to express weighted term search instructions. Thus, given any set of index term weights and a threshold, it is possible to derive an equivalent BRS using algorithms presented here. Others have attempted to specify index term weights which would simulate a given subjective BRS[13,14]. This is the inverse of the approach taken here.

1.14 Methods Used

A utility prediction function for documents is constructed from the training set. This utility function is used, together with the user-specified threshold τ to retrieve documents from the file which are predicted to be relevant.

In the context of pattern recognition systems, the threshold utility prediction function is a decision function. Each document in the system is represented as a vector $\underline{x}$ of index terms which is then assigned to one of two mutually exclusive categories, 'relevant' or 'non-relevant' by applying the decision function $[f(\underline{x}) - \tau]$.

The training set is submitted by the user. Each document in this set is assigned a utility on a pre-determined scale. Both relevant and non-relevant documents are represented. Feature extraction (dimensionality reduction) is first performed on training set vectors to reduce their dimensions. A subset of index terms is selected using an information theoretic measure. This measure gives an estimate of how well individual index terms discriminate between relevant and non-relevant documents in the training set.

Next a linear decision function is estimated using the reduced (in size) training set vectors. (For this application, $f(\underline{x})$ is a linear utility prediction function (LUPF).) Parameters in this linear model are estimated from the training set using the L_1 norm criterion of best approximation. This estimation problem is set up as a linear program and solved using the simplex algorithm.

Finally, by applying the LUPF to documents not in the training set, it is possible to identify all the documents in the file which are predicted to be relevant.

This identification can be done in two ways. By evaluating $f(\underline{x})$ for each $\underline{x}$ and comparing this to the threshold τ, each $\underline{x}$ may be classified individually. This method is appropriate for searching a sequentially structured file (SSF). An alternate method is to solve the linear pseudo-Boolean inequality (LPBI), $f(\underline{x}) \geq \tau$, for its solution families. This gives Boolean combinations of index terms which are the analytically derived BRS. The BRS form of the LUPF is necessary for searching an inversely structured file (ISF).

The BRS derived above is a set of matching templates which can be placed over a pattern vector $\underline{x}$ to categorize it. Each template corresponds to a solution family of the LPBI. Solution families to the LPBI are obtained using a branch-and-exclude binary tree search algorithm. Fig. 1-1 shows a block diagram of the system.

1.15 Testing and Results

Training sets were prepared for several test questions. Using these training sets, BRS's were written both automatically by the system and by a group of experienced NASA system users. A portion of the NASA file was searched using each of the BRS's.

Relevant documents had been identified beforehand and a measure of effectiveness was developed for each search which used this fact.

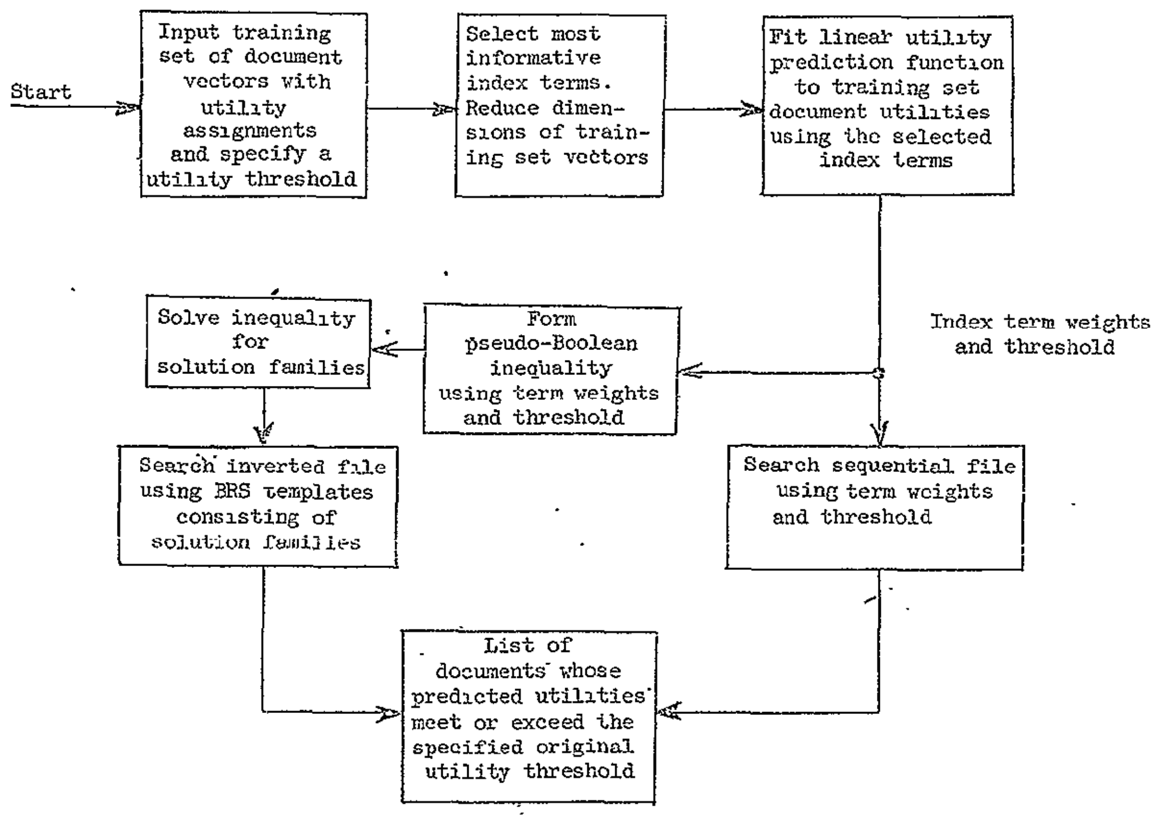
Test results showed that the machine-derived BRS's were only about half as effective as the subjective user-derived BRS's. Differences appear to be largely attributable to the use (by humans) of supplementary information not contained in the training set.

1.16 Conclusions

It is concluded that the pattern recognition model of document retrieval employed here is very useful for deriving an analytical BRS. However, more work is needed to increase the practical effectiveness of the automatic system, particularly in the area of feature extraction.

FIGURE 1-1

BLOCK DIAGRAM OF A DOCUMENT RETRIEVAL SYSTEM USING PATTERN RECOGNITION TECHNIQUES



1.2 Structure and Assumptions of the Model

The analytical assumptions made to model the process are listed and discussed below.

1.21 Document Representation and File Structure

A file of indexed documents is assumed to exist. Each document d_k, $k=1,2,\ldots,D$ in the file is represented as a binary vector $\underline{x}_k = (x_{ik})$, $i=1,2,\ldots,\Omega$, of index terms, chosen from a master list having Ω terms. If index term i is assigned as a characteristic to document d_k, then $x_{ik} = 1$. Otherwise $x_{ik} = 0$. For example, in the NASA system, $\Omega \cong 13{,}000$; $D \cong 500{,}000$ and about eleven $x_{ik} = 1$ for each k.

The entire file may be conveniently pictured as a <u>binary document-term</u> matrix having Ω rows and D columns. Each row index corresponds to an index term T_i, where all terms are arranged in some standard order (such as alphabetically) and each column index k corresponds to a document number n_k, where all document numbers are also arranged in some standard order (such as chronologically). Because the matrix is very sparse, it is convenient to represent it in a more compact form. There are two ways to readily do this by collapsing either the matrix columns or rows.

To collapse the matrix columns, represent each column (document) vector $\underline{x}_k$ as a list L_k of row indices $L_k = (\ell_{k1},\ldots,\ell_{k\Gamma_k})$ having Γ_k members. Here ℓ_{jk} are row indices corresponding to

$x_{jk} = 1$. The list L_k simply identifies the index terms used with a given document. For example, with the NASA system there would be about 500,000 lists having an average of 11 members each. A data structure can now be defined having a master list of document numbers n_k; $k=1,2,\ldots,D$ where each n_k has an associated sub-list L_k of index term numbers. This data structure will be defined as a sequentially structured file (SSF).

Alternately, it is possible to collapse the matrix rows. Each row can be represented as a list C_i of column indices having Δ_i members. $C_i = (c_{i1},\ldots,c_{i\Delta_i})$, where c_{ij} are column indices corresponding to $x_{ij} = 1$. This list identifies the documents associated with the index term T_i. The corresponding data structure has a master list of index terms, with each term having an associated sub-list of document numbers. This data structure is defined as an inversely structured file (ISF).

Observe that to locate in an SSF all d_k with $f(\underline{x}_k) \geq \tau$ it is necessary to examine every list L_k, form $f(\underline{x}_k)$ from this list and then make a decision.

With an ISF, searching is done only with specified Boolean combinations of index terms (the BRS). Appropriate set operations on the lists C_i associated with the terms T_i will give a resultant set of document numbers. Since usually only a small subset of all T_i are specified in the BRS, the search of an ISF is more economical than the search of an SSF. The conversion of the condition $f(\underline{x}_k) \geq \tau$ to an equivalent BRS allows the more economical ISF search to be substituted for the SSF search. Given a file, it is easy to convert it from an ISF to an SSF or vice versa. We will represent a file of index terms and document numbers in <u>either</u> form as $F(X,T_i,n_k)$.

1.22 Fundamental Assumptions

1.221 <u>File Existence</u>. A file $F(X,T_i,n_k)$ of indexed documents $d_k, k=1,\ldots,D$ exists. The $n_k, k=1,2,\ldots,D$ are document numbers, while the $T_i, i=1,2,\ldots,\Omega$ are index terms.

1.222 <u>Document Utility</u>. Each document d_k represented in the file has a personal utility u_k to a given user at a given time. The utilities u_k can be measured on an arbitrary scale.

1.223 <u>Document Relevance</u>. A threshold τ (dependent on the chosen utility scale) can be specified by a user to define relevant and non-relevant documents. ($u_k \geq \tau \Rightarrow d_k$ is relevant).

1.224 <u>System Objective</u>. The objective of the system is to provide a list from the file F of document numbers n_k, corresponding to all relevant d_k.

1.225 <u>Source of Information for Utility Prediction</u>. The utility u_k of any document d_k may be adequately predicted as some function of $\underline{x}_k$, where $\underline{x}_k$ is the column vector of X associated with document d_k, i.e., $u_k = f(\underline{x}_k)$. This assumption <u>disallows</u> the use of information which is not associated with the document characteristics in the file.

1.226 <u>Dimensionality Reduction</u>. For the purposes of any given user, all but a small subset of all index terms may be neglected without a significant loss of information. This allows the vectors $\underline{x}_k$ to be reduced in dimension.

1.227 Linear Utility Function. A prediction of document utility $\hat{u}_k$ is adequately given by

$$\hat{u}_k = f(\underline{x}_k) = \beta_o + \sum_{j=1}^{n} x_{jk}\beta_j .$$

1.228 Estimation of Parameters in the Linear Utility Function. The parameters $\beta_j, j=0,1,\ldots,n$ in the linear utility function may be adequately estimated from examples in a training set of m documents where $m > n$.

1.3 Limitations

Assumptions 1.221 through 1.225 are rather general. Assumption 1.225 implies that the quality of indexing is adequate for the group of users who will retrieve from the file.

Assumption 1.226 is quite restrictive since it assumes that all but a small set of index terms may be discarded without a significantly degrading system performance. This of course is always done by users who form a BRS with only a few (from 3 to 15) index terms selected subjectively from the master list. This same assumption is also made frequently in pattern recognition systems design, where it is termed 'pre-processing' or 'feature extraction'. It is also numerically necessary to reduce the size of the vectors $\underline{x}_k$ before continuing with the estimation problem of assumptions 1.227 and 1.228.

Assumption 1.227 assumes a linear utility function for convenience in estimating the parameters. This is a fairly restrictive assumption.

Assumption 1.228 implies that the sample adequately represents users interests over the entire file. The number of documents m in the training set must be greater than or equal to the parameters β_j which are estimated. This relates to assumption 1.226, since the final reduced dimension of the training set vectors fixes the maximum number of parameters which may be estimated.

1.4 Organization of this Dissertation

This dissertation is presented in nine chapters, which describe system design and tests performed on the NASA file.

Chapter 2 describes a simple pattern recognition system, but not in the context of document retrieval. An example problem illustrates system operation. Example patterns are classified using both the linear decision function and the matching templates which are derived from it, by solving a pseudo-Boolean inequality.

Chapter 3 relates the system of chapter 2 to a similar system for the document retrieval problem. Document utility is defined and measured on an arbitrary scale. A user specified threshold is introduced on this utility scale to define relevance. The decision function can now be interpreted as a utility prediction function. The matching templates for classifying patterns are shown to be identical in form and use to the subjective BRS.

Chapter 4 develops the information theoretic measure for extracting best index terms as an extension of decision theory when utilities for action-outcome pairs are not known. This information theoretic measure has been used in other recognition systems for extracting pattern features. See, for example, Lewis[15] and Maltz[16]. The interpretation here is different and follows Watanabe[17] more closely.

Chapter 5 illustrates the determination of index term weights by using approximation theory. The L_1 norm problem is formulated as a linear programming problem (see Barrodale[18,19]). Examples are given illustrating alternate optimal solutions. Special properties of the solution are noted.

Chapter 6 presents the theory of pseudo-Boolean inequalities as developed by Hammer and Rudeanu[20,21,22]. A composite algorithm is presented here which solves a pseudo-Boolean inequality by a branch-and-exclude technique carried out in the context of a binary tree search. The basic branch-and-exclude technique is that developed by Hammer and Rudeanu. To implement this technique, a binary tree traversal[23] subalgorithm is introduced which controls and sequences the tree search. The composite algorithm is called the Tree Pruning[1]

---

[1]This name was used by E.W. Kozdrowicki[27] to describe a general process of branching and excluding in operations with tree structures. Because of the accurate description which it also conveys about the operations of solving a pseudo-Boolean inequality, it is used again here.

Algorithm (TPA). An example problem is solved and computational experience with the TPA is discussed.

Chapter 7 describes system testing which is carried out by using a 2^3 factorial design. The main factor tested was the difference in the effectiveness of searches performed using BRS's subjectively derived by analysts and BRS's analytically derived by the methods of chapter 3. Three measures of effectiveness were used to evaluate search effectiveness. The more traditional measures of recall and precision were both used[24,25]. In addition an information theoretic measure suggested by Meetham[26] was used. Other factors tested were those of training set size and the number of extracted features.

Chapter 8 discusses results of the testing, and presents ancillary data felt to be of interest. Searches done using subjective BRS's were significantly more effective than those performed using the analytically derived BRS's. The difference is largely attributable to a significant difference in <u>precision</u> of subjective and machine searches. This difference in precision seems related to the human use of information <u>not contained</u> in the training set. The extra information allows human analysts to <u>avoid</u> using index terms which have a high frequency of occurrence, even though they are excellent discriminators over the training set.

Chapter 9 suggests improvements and extensions of some of the concepts which appear useful. The generality of the pattern recognition model is apparent from the number of possible extensions.

Appendix A provides an example of the processing of a typical document training set to produce a BRS. Programs were written in Fortran IV for the IBM 7094/7044 Direct Couple System.

It is concluded that the pattern recognition model presents a very convenient analytical framework to use for document retrieval system analysis and design. Resolution of significant differences between automatic systems and human beings appears to be within the realm of possibility if more sophisticated automatic systems are designed.

1.5 Contributions

The contributions of this dissertation are felt to be in three areas: models, methods, and data.

1.51 Models

Modeling the derivation of the BRS as a pattern recognition problem is felt to be significant because it allows rigorous analytical methods developed by others (information theory, approximation theory, linear programming) to be applied directly to the document retrieval problem. This is an application of existing technology to a new area.

The conversion of a linear decision function to equivalent matching templates by solving an associated LPBI is a new application of pseudo-Boolean programming to pattern recognition systems.

The analogy between the BRS of document retrieval systems and the matching templates of pattern recognition systems makes this new template-generation technique immediately applicable to document retrieval systems utilizing inversely structured files (ISF's).

1.52 Methods

Generation of matching templates by solving an LPBI for its solution families is made practical by development of an algorithm to carry out the required computations quickly and efficiently. No claim is made here to the general method of LPBI solution via branch-and-exclude operations in a binary tree. This is due to Hammer and Rudeanu. The contribution here is the adaptation of a sub-algorithm to efficiently organize and sequence the branch-and-exclude operations.

1.53 Data

Testing of the model and methods on the NASA document retrieval system has given new data on which to plan future system modifications and retrieval experiments.

In addition, a limited amount of data is also available on operation of the TPA (tree pruning algorithm) for solution of the LPBI. This data should provide a basis for comparison of the present TPA with future modified versions as they are developed.

2.0 PATTERN RECOGNITION SYSTEMS

This chapter introduces and briefly describes a pattern recognition system of the type which will be applied to the document retrieval problem.

The general concepts of feature extraction, decision function formation and template matching operations are introduced and discussed. One simple example is used throughout the chapter to illustrate these concepts.

2.1 Introduction

Pattern recognition systems are concerned with the automatic classification of patterns (represented as vectors) into two or more mutually exclusive categories. A training set of pre-classified patterns is assumed available to 'train' the recognition system. After 'training', patterns of unknown classification are presented to the recognition system. If the training set was 'typical' in some sense, then the recognition system should classify the unknown patterns 'reasonably well'.

The simplest pattern recognition system is one which works with binary pattern vectors $\underline{x} = (x_1, x_2, \ldots x_n)$ where $x_i \varepsilon \{0,1\}$, and classifies all patterns into one of two categories. This is the type of system to be considered here. For general references to the subject of pattern recognition, see for instance Nilsson[28], Nagy[29] or J. Ho[30].

2.2 Pre-Processing

Training of a recognition system can be considered in two parts. The first part concerns representation of pictures or other patterns as vectors, and will be called pre-processing[31]. The second part estimates parameters of a decision function from vectors in the training set.

2.21 Representation of the Pattern as a Vector

Figure 2-1A illustrates a group of 5 simple patterns. A recognition system is desired which will distinguish between binary patterns representing pictures of the letters A and B. Let these patterns become the training set, which contains two 'pictures' of the letter A and three of the letter B. The grids of the pictures shown are 4 x 4. If we agree to order the rectangular sub-elements of the pictures from left to right and from top to bottom, then we can represent each picture of Fig. 2-1A as a binary vector $\underline{x}_k$ as shown in Fig. 2-1B, where $x_{ik} = 1$ if any element of the k^{th} picture of A or B lies within the i^{th} rectangle and $x_{ik} = 0$ otherwise.

2.22 Feature Extraction

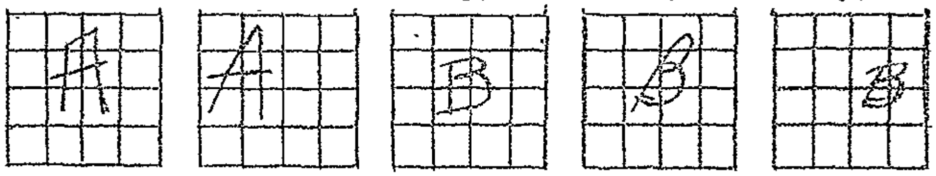
The next step in designing an automatic recognition system is usually to reduce the dimension of the pattern vectors by discarding vector elements which are 'non-informative'. This operation is also known as 'feature extraction'. It is a very important portion of the pre-processing operation. Heuristically we can see that vector

FIGURE 2-1

EXAMPLE SHOWING PATTERNS REPRESENTED AS VECTORS AND ILLUSTRATING FEATURE EXTRACTION

A. Training Set of Patterns P_k, $k=1,2,\ldots,5$

1. 2. 3. 4. 5.



B. Pattern Vectors $\underline{x}_k$, $k=1,2,\ldots,5$

$\underline{i}$	$\underline{x}_1$	$\underline{x}_2$	$\underline{x}_3$	$\underline{x}_4$	$\underline{x}_5$
1	0	0	0	0	0
2	1	1	0	0	0
3	1	0	0	1	0
4	0	0	0	0	0
5	0	1	0	0	0
6	1	1	1	1	0
7	1	0	1	1	1
8	0	0	0	0	1
9	0	1	0	0	0
10	1	1	1	1	0
11	1	0	1	1	1
12	0	0	0	0	1
13	0	0	0	0	0
14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0

Ordering of Grid Points

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

C. Feature Vectors $\underline{z}_k$, $k=1,2,\ldots,5$

$\underline{i}$	$\underline{z}_1$	$\underline{z}_2$	$\underline{z}_3$	$\underline{z}_4$	$\underline{z}_5$	
1	1	0	0	1	0	$z_{k1} = x_{k3}$
2	0	1	0	0	0	$z_{k2} = x_{k5}$
3	0	0	0	0	1	$z_{k3} = x_{k8}$

$$\left.\begin{array}{l} z_{k1} = x_{k3} \\ z_{k2} = x_{k5} \\ z_{k3} = x_{k8} \end{array}\right\} k=1,2,\ldots,5$$

elements 1,4 and 13-16 contain no information at all, since they are always zero, regardless of whether the pattern is an A or a B. Vector element 2 is a perfect classifier of the patterns in the training set, since $x_{2k} = 1$ when $k=1,2$ (letter A) and $x_{2k} = 0$ for $k=3,4,5$ (letter B). Vector elements 3,5 and 6-12 give some information about the correct classification of the vectors even though they are not perfect predictors.

The notion of information content over the training set can be formalized by using the concept of entropy from information theory. This will be done later. Assume for illustrative purposes that all vector elements except 3,5 and 8 have been discarded. Then elements 3,5 and 8 represent 'features' which have been extracted by the information screening process. The resulting 5 three-dimensional feature vectors $\underline{z}_1,\ldots,\underline{z}_5$ are shown in Fig. 2-1C. Note that $z_{k1} = x_{k3}$, $z_{k2} = x_{k5}$ and $z_{k3} = x_{k8}$ for $k=1,\ldots,5$.

2.3 Decision Function Specification

The second major step in the machine training process is to specify a decision function. This function is given as $y = f(\underline{z})$. It maps the feature vectors $\underline{z}$ of patterns of unknown classification into the dependent variable y on the real line.

The form of the function $f(\underline{z})$ is specified while the parameters of $f(\underline{z})$ are estimated from the training set.

The decision function $f(\underline{z})$ is used as follows. Assume a pattern vector $\underline{x}$ of unknown classification is to be put into category

A or B. First, vector $\underline{x}$ is reduced to vector $\underline{z}$ by extracting the features selected as being 'informative' over the training set. Then $f(\underline{z}) = \hat{y}$ is computed and if $\hat{y} \geq \tau$ (a given threshold), then the vector $\underline{z}$ (or $\underline{x}$) is assigned to category A. If $\hat{y} < \tau$, then $\underline{z}$ is assigned to category B.

2.31 Selecting the Form of Decision Function

There are two methods generally used to select the form of $f(\underline{z})$. If the vectors $\underline{z}$ are from a known multivariate probability distribution $p(\underline{z})$, then the form of $f(\underline{z})$ may be derived from the form of this distribution. The parameters of $p(\underline{z})$ which appear in $f(\underline{z})$ will be estimated from the training set. This is known as parametric decision function formation.

The other method used to specify $f(\underline{z})$ is known as nonparametric decision function formation. Here the form of $f(\underline{z})$ is chosen as a matter of convenience, and the parameters are estimated from the training set samples. Nonparametric methods are used exclusively for the applications to be considered here.

A very convenient form for the decision function and the one to be considered here is the linear function

$$y = L(\underline{z}) = \beta_0 + \sum_{j=1}^{n} \beta_j z_j$$

$$-\infty < \beta_j < \infty$$

$$z_j \varepsilon \{0,1\} ;$$

The β_j are the feature weights, while the z_j are binary elements of the feature vector $\underline{z}$.

2.32 Estimating the Parameters of the Decision Function

The parameters β_j are estimated by an approximation process from the samples in the training set. If the training set is large and typical of the universe of unknown patterns to be classified, then good results should be expected when $\hat{y} = L(\underline{z})$ is used to classify unknown patterns.

2.321 The Associated Approximation Problem. There is considerable freedom in choosing a method of estimating the β_j. Nearly all methods involve the choice of a best approximation to the β_j based on the training set. This type of problem has been studied extensively by mathematicians, to whom it is known as the discrete linear approximation problem[32,33]. Consider the following relationships for a training set of n pattern vectors having $m < n$ elements each:

$$y_i = \sum_{j=0}^{m} \beta_j z_{ij} + r_i \; ; \qquad i=1,2,\ldots,n; \; z_{io} \equiv 1;$$

or

$$\underline{y} = Z\underline{\beta} + \underline{r} \; .$$

Here $\underline{y}$ is an (nx1) vector of known binary variables obtained from the training set: $y_i = +1$ if pattern i belongs to category A and

$y_i = -1$ otherwise. $\underline{\beta} = (\beta_j)$ is an $(m \times 1)$ vector of unknown parameters (feature weights). $Z = (z_{ij})$ is a known matrix $(n \times m)$ of binary variables obtained from the training set. Rows of Z are the training set feature vectors $\underline{z}$. The unknown vector of residuals $(n \times 1)$ is denoted by $\underline{r} = (r_i)$.

The problem is to estimate $\underline{\beta}$. Call this estimate $\underline{b}$ (note that $\underline{\beta}$ can never be known exactly as long as the training set is only a sample of the universe of all patterns $\underline{z}$). Note that $\hat{\underline{y}} = Z\underline{b}$ is an estimate of $\underline{y}$ based on the estimate $\underline{b}$ of $\underline{\beta}$. Then $\underline{r} = \hat{\underline{y}} - Z\underline{b} = \underline{y} - \hat{\underline{y}}$.

2.322 Choosing the Criterion of Best Approximation. By a best estimate $\underline{b}$ of $\underline{\beta}$ we shall mean the vector $\underline{b}$ which minimizes the length (norm) $||\underline{r}||$ of the vector $\underline{r}$. There are many ways of specifying a norm. An entire class of norms is given by the L_p (L sub p) norms defined below:[34,35]

$$L_p(\underline{r}) = \left(\sum_{i=1}^{n} |r_i|^p\right)^{1/p} = ||\underline{r}||_p$$

When $p = 2$, $L_p(\underline{r}) = \left(\sum_1^n r_i^2\right)^{1/2}$ and we get the familiar least squares problem. When $p = 1$, we have $L_1(\underline{r}) = \sum_{i=1}^{n} |r_i|$, and in the limit as $p \to \infty$ we have $L_\infty(\underline{r}) = \max_i |r_i|$. This is also known as the Chebyshev, uniform or max norm. The approximation problem can now be written as follows. Find $\underline{b}$, such that

$$\underline{b} = \min_{\underline{r}} ||\underline{r}||_p = \min_{\underline{a}} ||\underline{y} - Z\underline{a}||_p .$$

All practical applications of discrete approximation theory known to the author use either the L_1, L_2 or L_∞ norms (or some variation of them), since these three formulations have solution algorithms which are reasonable to implement on a computer. Most applications utilize the L_2 norm. The solution for $\underline{b}$ is given then by the familiar least squares normal equations[36].

$$\underline{b} = (Z'Z)^{-1}Z'\underline{y}$$

Both the L_1 and L_∞ norm problems can be cast as linear programming (LP) problems, which are readily solvable by the simplex algorithm or one of its variations[37,38,39].

The popularity of the L_2 norm is due largely to the following items:

(a) familiarity of the method, and of the solution algorithms;

(b) statistical applications of least square estimators when the r_i are normally distributed[40]; and

(c) uniqueness of the solution vector $\underline{b}$.

Least squares estimation has the <u>disadvantage</u> that the $n \times m$ matrix Z must have all m columns linearly independent to insure that $(Z'Z)$ will be nonsingular.

The L_1 and L_∞ estimators of $\underline{\beta}$ have the following characteristics:

(a) ease of solution when formulated as LP problems;

(b) the $n \times m$ matrix Z is not required to have all m columns linearly independent to guarantee a solution;

(c) the L_1 and L_∞ estimators can be better estimators of $\underline{\beta}$ than L_2 when the r_i are not normally distributed[41];

(d) L_1 and L_∞ estimators are not necessarily unique. The same minimal value of $L(\underline{r})$ can be attained for more than one solution vector $\underline{b}$[42].

The overall differences in estimates of $\underline{\beta}$ based on L_1, L_2 and L_∞ norms can be negligible. Choice of a norm for applied problems often depends upon practical considerations.

In the application to document retrieval systems to be presented in chapter 3, the L_1 formulation will be utilized for the following two reasons:

(a) the columns of Z cannot be guaranteed independent so that further checking would be required if the L_2 norm were used.

(b) the L_1 problem is very rapidly and efficiently solved in the linear programming (LP) formulation.

2.33 Current Methods of Forming Decision Functions

A great number of pattern recognition decision functions are linear. Several techniques for estimating the parameters are based on methods which are variations of the L_1 or L_∞ norm. See for example Smith[43] or Grinold[44]. Least squares methods are also used. See for instance Y.C. Ho[45]. For another formulation less recognizable as an approximation problem, see Mangasarin[46] or Taylor[47].

2.34 An Example Problem Illustrating Decision Function Determination

In the example used to illustrate feature extraction, features 3,5 and 8 were arbitrarily chosen, and the feature vectors $\underline{z}_1 \ldots, \underline{z}_5$ were formed. These vectors now represent the training set, instead of the vectors $\underline{x}_1, \ldots, \underline{x}_5$.

Figure 2-2A shows the model $\underline{y} = Z\underline{\beta} + \underline{r}$ for this example. The least squares criterion is used to derive a solution $\underline{b}$ as shown in Fig. 2-2B. The least squares solution is used for this example problem only. All subsequent problems will use the L_1 norm criterion. The residual vector for the least squares solution is shown in Fig. 2-2C.

In Fig. 2-2A the $n = 5$ rows of the matrix Z are the n feature vectors $\underline{z}_1, \ldots, \underline{z}_5$ which constitute the training set for the problem. Each vector $\underline{z}_k$ is augmented by adding unity in the first position.

The columns of the matrix Z (excluding the first column) correspond to the 0/1 'features' which were extracted from the original training set vectors $\underline{x}$. The first column is a vector of all 1's which is included to allow a constant term in the decision function.

FIGURE 2-2

EXAMPLE SHOWING LINEAR DECISION FUNCTION PARAMETER ESTIMATION

A. Linear Model Estimation

$$\underline{y} = Z\underline{\beta} + \underline{r}$$

$$\begin{bmatrix} +1 \\ +1 \\ -1 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_o \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{bmatrix}$$

B. Least Squares (minimal $L_2(\underline{r})$) Solution for $\underline{b}$, the Estimator of $\underline{\beta}$

$$\underline{b} = \min_{\underline{\beta}} ||\underline{y} - Z\underline{\beta}||_2 = (Z'Z)^{-1}Z'\underline{Y} = \begin{bmatrix} -1 \\ 1 \\ 2 \\ 0 \end{bmatrix}$$

$$\hat{y} = -1 + 1z_1 + 2z_2 + 0z_3$$

$$\tau = 0$$

C. Residual Vector for Least Squares Estimator

$$\underline{r} = \underline{Y} - \hat{\underline{Y}} = \begin{bmatrix} +1 \\ +1 \\ -1 \\ -1 \\ -1 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} +1 \\ 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

$$||\underline{r}||_2 = \sqrt{1 + 1} = \sqrt{2}$$

The vector $\underline{y}$ of dependent variables consists of the elements $y_i = +1$ or -1 where $y_i = +1$ is used for patterns of letter A in the training set and $y_i = -1$ is used for patterns of letter B.

The problem is specified completely when τ is chosen. The threshold τ is used for making decisions after $\underline{\beta}$ is estimated. This threshold is somewhat arbitrarily specified as the midpoint 0 between $y_i = +1$ and $y_i = -1$. If $\hat{y} \geq \tau = 0$ for some unclassified pattern, then we agree to decide that this pattern $\underline{z}$ represents the letter A and if $\hat{y} < \tau$, then $\underline{z}$ represents B.

Fig. 2-2B shows the least squares solution $\underline{b} = (-1,1,2,0)$. Here the feature z_3 has been assigned a weight zero ($b_3 = 0$).

The results of applying the model to the training set as a predictor are given in Fig. 2-2C, which compares y and $\hat{y}$. Here the two A patterns are correctly classified, but one of the B patterns (pattern 4) is misclassified or rejected since $\hat{y}_4 = 0$. The linear relationships $\hat{\underline{y}} = Z\underline{b}$ is thus not completely adequate to correctly classify all the documents in the training set.

There is information lost at two points. First, the feature extraction process throws away information by discarding potentially important features. Secondly, the approximate linear decision function may introduce errors. Perhaps a better decision function would be nonlinear. Or perhaps the training set should be larger.

The fact that any pattern recognition system will make errors must be accepted; although it must be trained to have a minimal (often zero) error for the sample patterns. The emphasis is on picking a reasonable system design and then adjusting it so that its recognition error rate is acceptable for the application at hand.

2.35 Relationship of the Decision Function to Curve-Fitting Problems

The standard curve-fitting or regression model is given by[48,49]

$$\underline{y} = Z\underline{\beta} + \underline{r}$$

and is identical to the decision function model. The difference is entirely in interpretation. In ordinary function fitting applications the dependent variables $\underline{y}$ are the yield of some process. In the pattern recognition problem, the y_i are fixed at ± 1, to indicate two different categories.

One way of resolving the apparent difference between the two is to regard the y_i as the differences between two probabilities

$$y_i = p(A/\underline{z}) - p(B/\underline{z})\ .$$

Then since $p(A/\underline{z}) = 1$ and $p(B/\underline{z}) = 0$ or vice versa for all training patterns in categories A or B, it follows that

$$(p(A/\underline{z}) - p(B/\underline{z}))\varepsilon\{-1,+1\}.$$

If we agree to assign patterns to category A when $\hat{y} \geq \tau = 0$ we see that;

$$\hat{y} = p(A/\underline{z}) - p(B/\underline{z}) \geq \tau = 0$$

$$\Longrightarrow p(A/\underline{z}) \geq p(B/\underline{z}) \Longrightarrow \frac{p(A/\underline{z})}{p(B/\underline{z})} \geq 1.$$

Thus by assigning patterns to category A when $\hat{y} \geq \tau = 0$ we are making a reasonable decision based on estimated probabilities. This explanation of the decision function can be called the "potential function" interpretation[50].

The independent variables z_{ij} in the problem are binary. In the statistical literature linear least squares models of this type are referred to as "experimental design models".[51]

2.4 Template Matching Operations

2.41 Introduction

Once the decision function is determined, the category of any unclassified pattern $\underline{x}$ may be estimated by first converting $\underline{x}$ to $\underline{z}$ then by forming $\hat{y} = b_o + \sum_{1}^{m-1} b_j z_j$ and comparing this with the threshold zero.

There is an alternative to computing $\hat{y}$ and comparing it to a threshold. This is the formation of groups of one or more templates which compare specified combinations of binary features in the original pattern vectors $\underline{x}$, or in the feature vectors $\underline{z}$.

2.42 The Pseudo-Boolean Inequality

The mathematical motivation behind this comes from the theory of pseudo-Boolean inequalities (binary variables and real coefficients). Note that:

$$\hat{y} \geq \tau \Longrightarrow b_o + \sum_{j=1}^{n} b_j z_j \geq \tau$$

$$\Longrightarrow \sum_{j=1}^{n} b_j z_j \geq (\tau - b_o) ,$$

which is a pseudo-Boolean inequality.

Binary vectors $\underline{z}$ are mapped onto the real line via the real coefficients b_j. All binary vectors $\underline{z}$, which satisfy the inequality are solution vectors. Each solution vector represents a binary pattern vector $\underline{z}$ which belongs to category A. The solution vectors $\underline{z}$ can be grouped and placed into one or more solution families. Each solution vector belongs to one and only one family.

The families specify a fixed configuration of either 0 or 1 for some of the variables in the vector, and a free configuration for others.

To illustrate how solution vectors may be grouped into families, consider some hypothetical inequality with six solution vectors $\underline{z} = (z_1,z_2,z_3,z_4)$ and two solution families.

$$F_1(\underline{z}) = (1,0,-,0) \Longrightarrow \begin{cases}(1,0,1,0)\\(1,0,0,0)\end{cases}$$

$$F_2(\underline{z}) = (-,1,1,-) \Longrightarrow \begin{cases}(1,1,1,0)\\(1,1,1,1)\\(0,1,1,0)\\(0,1,1,1)\end{cases}$$

All 6 solution vectors lie in either family $F_1(\underline{z})$ or family $F_2(\underline{z})$. $F_1(\underline{z})$ is a compact representation of 2 solution vectors while $F_2(\underline{z})$ represents 4 solution vectors. Another way of writing the families is $F_1(\underline{z}) = z_1\bar{z}_2\bar{z}_4$; $F_2(\underline{z}) = z_2z_3$.

Families of solutions may be regarded as <u>matching templates</u> for the patterns $\underline{z} = (z_1,z_2,\ldots,z_4)$. For example, $F_2(\underline{z})$ requires the <u>simultaneous</u> presence of a 1 in components 2 and 3 of the vector $\underline{z}$. All vectors $\underline{z}$ with a 1 in both components 2 and 3 will match the template $F_2(\underline{z})$. Similarly all vectors with a 1 in position 1 and 0's in both positions 2 and 4 will match the template $F_1(\underline{z})$.

In this example all <u>solution</u> vectors belong to either family $F_1(\underline{z})$ or $F_2(\underline{z})$. Also, all solution vectors $\underline{z}$ satisfy the thresholded linear decision function given by

$$\sum_{j=1}^{n} b_j z_j \geq \tau - b_o .$$

It follows that all pattern vectors $\underline{z}$ which match either template $F_1(\underline{z})$ or $F_2(\underline{z})$ belong to category A $(\hat{y} \geq \tau)$ and all patterns which <u>fail</u> to match either template belong to category B $(\hat{y} < \tau)$.

It is convenient to define the <u>characteristic</u> <u>function</u> of a pseudo-Boolean inequality as a matching operation on the <u>union</u> of all solution families (templates) $F_k(\underline{z})$, $k=1,2,\ldots, M$.

$$\phi(\underline{z}) = \bigcup_k [F_k(\underline{z})]$$

$\phi(\underline{z})$ is a <u>Boolean</u> function which takes on the value of 1 when the pattern vector $\underline{z}$ matches one of the M templates and takes on the value 0 when a match does not occur.

It follows that

$$\phi(\underline{z}) = 1 \Longrightarrow \underline{z} \text{ belongs to category A ;}$$

$$\phi(\underline{z}) = 0 \Longrightarrow \underline{z} \text{ belongs to category B .}$$

Observe that the solution of the pseudo-Boolean inequality derived from the thresholded decision function involves no approximation process. No information is lost. The matching templates for making binary decisions about the classification of patterns are merely an alternate form of implementing the decision function. Instead of adding weights for vector elements which are present and comparing the sum to a threshold, we look instead for the presence of configurations of points. If one of the configurations is observed, we automatically assign the pattern to category A. For some recognition systems this matching of configurations is a more effective method of identifying patterns. Families of solutions to a linear pseudo-Boolean inequality may be found by a branch-and-exclude binary tree search algorithm.

2.43 An Example of Classification by Template Matching

The example problem considered previously in this section has an associated pseudo-Boolean inequality

$$\hat{y} = b_o + \sum_{j=1}^{3} b_j z_j = -1 + 1z_1 + 2z_2 + 0z_3$$

$$\hat{y} \geq \tau \Longrightarrow \hat{y} \geq o \Longrightarrow \sum b_j z_j \geq (\tau - b_o)$$

$$1z_1 + 2z_2 \geq 1 .$$

This pseudo-Boolean inequality has two solution families:

$$F_1(\underline{z}) = (-,1) ;$$

$$F_2(\underline{z}) = (1,0) .$$

The family $F_2(\underline{z})$ has only one solution vector and is said to be <u>degenerate</u>. The characteristic function of the unequality is

$$\phi(\underline{z}) = (z_2) \cup (z_1 \bar{z}_2) .$$

Applying the $F_1(\underline{z}) = z_2 = x_5$ template to each of the 5 patterns in the training set (see Fig. 2-1A) gives a match for pattern 2. The $F_2(\underline{z}) = z_1\bar{z}_2 = x_3\bar{x}_5$ template gives a match for patterns 1 and 4. Thus patterns 1, 2 and 4 satisfy the characteristic function ($\phi(\underline{z}) = 1$) and are predicted to belong to category A. Pattern 4 is still incorrectly classified (see Fig. 2-2C).

2.5 Summary

This chapter has introduced and illustrated the principles involved in the design of a recognition system of the type to be used for the document retrieval problem. This is the two-category system using binary pattern vectors and a non-parametric linear decision function.

The steps involved in the design are:

(a) representation of patterns as vectors, and choice of a training set;

(b) feature extraction to reduce the pattern vector dimensions;

(c) specification of a linear decision function and estimation of the parameters in this linear function. Parameters are estimated from the training set with a discrete linear approximation model, and

(d) construction of templates from the decision function, using the pseudo-Boolean inequality. This gives an alternate (to the linear decision function) method of categorizing new patterns.

3.0 MODELING THE DOCUMENT RETRIEVAL PROCESS AS A PATTERN RECOGNITION SYSTEM

This chapter first describes a document retrieval system (DRS). Next an associated pattern recognition system is defined. The operations of characterizing the patterns, feature extraction, and decision function specification are related to the DRS. The implementation of the decision function to retrieve relevant documents from a file is presented in detail. Computer methods are briefly described.

3.1 The Document Retrieval System

3.11 General

The system to be described here is quite general. In fact, it is identical to the NASA document retrieval system[53,54]. This is a large system which has been in operation since 1962. Approximately 500,000 documents (technical reports and articles) are accessible through the system. A master list of about 13,000 index terms is used to index each document, with an average of about 11 index terms per document. A variety of services are available to users of this system. Computer searches are performed in both a batch processing[55] and a time-shared mode using remote terminals[56].

3.12 Representation of Documents in the File

Each document acquired by the retrieval system is assigned both a unique identification number and a set of index terms (index set) which are chosen from a master list. These index terms may in fact be phrases or word groupings which are deemed to have meaning to the users of the system.

All acquired documents are placed in a library, while their identification numbers and index terms form a unit record which is placed in a computer file.

3.13 Specification of File Search Instructions

The file is searched to identify documents which have specified combinations of terms in their index sets. These index term combinations are specified by the system users as intersections, unions and negations of index terms. The entire set of matching instructions is sometimes referred to as a Boolean retrieval strategy (BRS). A typical BRS is shown below:

((heat transfer + thermodynamic properties + thermal properties)
* (gases + gas flow)) - (fluid flow + fluid properties).

The symbol (+) is used for union (or), (*) is used for intersection (and), while (-) represents negation (but not). Parentheses are used where needed to avoid ambiguity. The BRS is specified subjectively by each user.

3.14 Satisfying User Needs

The computerized search system applies the BRS to the file and produces a list of document numbers.

Documents on this list match the BRS and may be recovered from the library. After looking at the actual documents (or abstracts of them) the user may elect to revise the BRS and search the file again. This can lead to an iterative type of search.

The user may elect to have an agent (called an information analyst) compose a BRS for him and screen the cited documents, rejecting those which do not (in the agent's opinion) match the user's interests. This practice relieves the user of the need to become familiar with operational details of the system, or with index term usage. A disadvantage is that the agent may misinterpret the user's interests.

Recent trends in the NASA DRS have been to introduce time-sharing facilities which permit direct user interaction with the file, and eliminate the need for an information analyst.

3.15 Problem Areas

There are numerous problem areas which can be associated with DRS's. Some of these are:

(a) poor search effectiveness;

(b) lack of a standard measure of search effectiveness;

(c) 'communication' difficulties between a human user and a computerized file;

(d) inadequate indexing; and

(e) lack of comprehensive analytical models for the above areas.

The alleviation of problem (c) above is the goal of this dissertation. A comprehensive analytical model is developed for the user-file communication process. The communication of the user with the file here refers to the formulation of search instructions by the user to specify how the file will be searched. It is assumed that an indexed file of documents exists, and also that a software system exists which will implement search instructions.

The present technique of subjectively selecting and combining index terms to form a BRS is very difficult. This difficulty is due to the large number of index terms, the extremely large number of ways to combine these terms and differences in word use between individuals (indexers and users). Each BRS which is subjectively formed requires solution of a difficult combinatorial problem.

The subjectively formed BRS now functions as the input to a file searching system. In the model introduced below, a BRS is provided as an end product. The user inputs information in the form of an example set of document numbers, with each document in the example set assigned a utility. In addition, each document is also assigned to one of two categories, relevant and non-relevant. This evaluated example set is all that is required of the user. The BRS formulation proceeds automatically using this information. None of the difficult combinatorial problems remain for the user.

The model used to automatically produce a BRS from an example set of documents is nearly identical with the pattern recognition system described in chapter 2. Details of model development are given below.

3.16 A Model of the Retrieval Process

Consider a file of indexed documents. Assume first that each document d_k in the file has a utility u_j (or measure of usefulness) to a given user at a given time. The utility of any given document can be determined by the user, and assigned a numerical value on some arbitrary scale (say 1 to 10). These are reasonable assumptions repeatedly used in operations research studies. See for example Fishburn[57] or Hadley[58].

Next assume that, dependent on the scale which is used to measure document utility that a threshold τ can be specified by the user which divides all documents in the file into two classes. Those documents d_k with $u_k \geq \tau$ are <u>defined</u> as being <u>relevant</u>. Those with $u_k < \tau$ are <u>not relevant</u>. The goal of the retrieval system is to retrieve all <u>relevant documents</u> and not retrieve any others.

3.2 An Associated Pattern Recognition System

3.21 Characterization of Documents as Pattern Vectors

Each document d_k can be represented as a binary vector $\underline{x}_k$. The elements of $\underline{x}_k$ are x_{kj}; $j=1,2,\ldots,\Omega$, where Ω is the number of index terms in the master list (about 13,000 for the NASA system). Each $x_{kj} = 1$ if index term j is used to index document k and 0 otherwise. On the average, only about 11 of the x_{kj} will be nonzero.

3.22 Definition of Two Categories

Each document d_k is either relevant or nonrelevant depending on whether its utility $u_k \geq \tau$ or $u_k < \tau$. These constitute the two mutually exclusive categories to which each document belongs. The function of the system will be to recognize relevant documents, or to assign documents to category A or B based on properties of the associated pattern vector $\underline{x}_k$.

Each user defines his own categories (relevant or not) depending on his personal utility for documents in the file. A training set is formed which represents a sampling of the personal utility function of an individual user. Thus, each user has an individual pattern recognition system at his disposal.

3.23 The Configuration of the System

The pattern recognition system designed to recognize relevant documents has the general configuration discussed below. (See also Fig. 1-1.)

3.231 Training Set Formation. The training set is composed of documents which have been selected by the user as being typically relevant or non-relevant. An estimate of the utility u_k of each document in the training set is provided by the user. Documents in the training set have been located via a manual search by the user, from a previous search, or from references provided by others. If the search is done iteratively, the training set grows and only the initial training set need be selected manually.

3.232 Feature Extraction. All index terms in the training set are ranked using an information-theoretic measure of goodness. This measure is the number of bits of information which each index term individually provides about the category of documents in the training set. Details are given in chapter 4. All index terms except a specified number with the highest information measure are discarded. The retained index terms are the 'extracted features'.

3.233 Decision Function Formation. The pattern recognition system of this chapter attempts to classify documents as relevant or not based on their predicted utilities. The categories are not absolutes, but are defined with reference to an arbitrary utility scale.

The system of chapter 2 was slightly different in character. Categories A and B there were absolute. Parameters in the decision function of chapter 2 were estimated by solving an approximation problem where the observed dependent variables y_i were dichotomous and could be regarded as the difference between two probabilities. The goal of the approximation problem was to 'best' approximate $y_i = p(A/\underline{z}) - p(B/\underline{z})$. The threshold was $\tau = 0$.

The decision function of the present chapter is also set up as an approximation problem, but the objective is to approximate the user assigned utilities of documents in the training set. The observed dependent variables y_i are no longer dichotomous and the threshold is now set by the user instead of being fixed at zero.

Another way of describing the differences in the decision functions is to consider the approximation model $\underline{y} = X\underline{\beta} + \underline{r}$. In chapter 2, the observed variables y_i are regarded as being fixed and non-random, while the matrix X is considered as a random variable. In this case variations in the residual vector $\underline{r}$ are caused entirely by variations in X.

In the system of this chapter, the observed y_i are regarded as random variables and the matrix X is fixed. Here the y_i are utility estimates which are corrupted by 'noise'. Variation in the residual vector $\underline{r}$ is caused entirely by variation in the observed variables y_i.

It can be seen that regardless of whether the matrix X or the vector $\underline{y}$ is taken to be the source of variability, that the model remains the same. In either case a reasonable estimate of $\underline{\beta}$ is one which minimizes the length of the residual vector $\underline{r}$. When the vector $\underline{y}$ is regarded as fixed, the decision function is often referred to as a discriminant function, and when the matrix X is fixed the decision function can be called an interpolation or regression function. The relation between approximation theory models and the pattern recognition process has been discussed by P.A.V. Hall[59].

The pattern recognition model used for document retrieval purposes here employs a linear decision function which is actually a regression function for predicting document utility as a function of 'extracted' index terms. The training set document utility estimates are regarded as noisy measurements. To emphasize this, the decision function of this model will be referred to hereafter as an LUPF (linear utility prediction function).

For reasons of convenience, the test configuration uses an integer utility scale where $y_i \varepsilon \{1,2,\ldots,9\}$ and τ is specified by the user. When $y_i = 1$, the document has no utility to the user and when $y_i = 9$, the document is most useful. The example problem presented later in this chapter uses a binary utility scale where $y_i \varepsilon \{+1,-1\}$. When $y_i = +1$ the document is relevant and when $y_i = -1$ the document is non-relevant. In this case the threshold $\tau=0$. Note that when this binary utility scale is used, that the LUPF here becomes identical to the decision function of chapter 2.

3.3 Implementing the Decision Function

3.31 Direct Method

Recall from section 2.4 that when a pattern vector $\underline{x}_k$ of unknown classification is to be assigned to either category A or B, there are two equivalent methods of making the decision by using the index terms in the decision function (the extracted features) which are common to the pattern vector $\underline{x}_{kj}$.

The direct method simply adds up the 'weights' of features in the vector $\underline{z}$ and compares the sum to the threshold, after which the vector $\underline{x}$ is put into the indicated category, i.e.,

$$\sum_{j=1}^{n} b_j z_j \geq (\tau - b_o)$$

implies that the pattern vector $\underline{x}$ is assigned to category A.

For the document recognition system, the index term weights are summed and compared to the utility threshold τ, after which the document vector $\underline{x}$ is classified.

3.32 Indirect Method

The indirect method derives matching templates by thresholding the decision function to form a linear pseudo-Boolean inequality (LPBI). This inequality is solved for its families of solutions. Details are presented in chapter 6. Each solution family becomes a matching template. If one of these templates matches the vector $\underline{x}$, then $\underline{x}$ is assigned to category A. Otherwise, $\underline{x}$ belongs to category B.

For the document recognition system, the matching templates correspond to combinations of index terms. Observe that <u>the matching templates are equivalent in form and function to the user's subjectively specified BRS</u>.

Thus, by considering document retrieval as a pattern recognition process, we <u>analytically</u> derive a BRS as a union of matching templates. This is an important result which allows the previously subjective BRS formation to be modeled as a feature extraction and decision operation.

To further illustrate this connection, consider the example BRS introduced in section 3.13. Figure 3-1 shows how this subjective BRS can be written as a union of solution families to some (unknown) pseudo-Boolean inequality (not necessarily linear, of course). Fig. 3-1A shows the original BRS. Fig. 3-1B shows the reduction of the BRS to a union of solution families. Fig. 3-1C shows the solution families in tabular form.

The solution families which result from reducing a <u>subjectively</u> determined BRS to the form of Fig. 3-1C are <u>not</u> necessarily mutually exclusive. For example, any documents containing the combination of index terms given by

$$\underline{T} = (T_1,T_2,T_3,T_4,T_5,T_6,T_7) = (1,0,0,1,1,0,0)$$

is covered by <u>both</u> solution families $F_1(\underline{T})$ and $F_2(\underline{T})$ shown in Fig. 3-1C. The solution families of an <u>analytically</u> determined BRS are mutually exclusive. This is important because no search effort is wasted by retrieving the same document with two different solution families.

3.33 Relation of Decision Function Implementation to Retrieval System File Structure

There are two basic methods of organizing computer files composed of index term - document number records. The first method is to have the document numbers arranged in a sequential master list in memory. Associated with each document number in this master list is a

FIGURE 3-1

EQUIVALENCE OF A SUBJECTIVE BRS TO A UNION OF SOLUTION FAMILIES

A. Subjective BRS

$$((T_1+T_2+T_3)*(T_4+T_5)) - (T_6+T_7)$$

WHERE: T_1= heat transfer
T_2= thermodynamic properties
T_3= thermal properties
T_4= gases
T_5= gas flow
T_6= fluid flow
T_7= fluid properties

B. Reduction of the Subjective BRS to a union of Solution Families

$$((T_1+T_2+T_3)*(T_4+T_5)) - (T_6+T_7)$$
$$= ((T_1*T_4)+(T_1*T_5)+(T_2*T_4)+(T_2*T_5)+(T_3*T_4)+(T_3*T_5)) - (T_6+T_7)$$
$$= (T_1*T_4*\overline{T}_6*\overline{T}_7)+\ldots+(T_3*T_5*\overline{T}_6*\overline{T}_7)$$
$$= (T_1T_4\overline{T}_6\overline{T}_7) \cup (T_1T_5\overline{T}_6\overline{T}_7) \cup (T_2T_4\overline{T}_6\overline{T}_7) \cup (T_2T_5\overline{T}_6\overline{T}_7) \cup (T_3T_4\overline{T}_6\overline{T}_7) \cup (T_3T_5\overline{T}_6\overline{T}_7)$$
$$= \left[F_1(\underline{T})\right] \cup \left[F_2(\underline{T})\right] \cup \left[F_3(\underline{T})\right] \cup \left[F_4(\underline{T})\right] \cup \left[F_5(\underline{T})\right] \cup \left[F_6(\underline{T})\right]$$

C. Solution Families in Tabular Form

	T_1	T_2	T_3	T_4	T_5	T_6	T_7
F_1	1	-	-	1	-	0	0
F_2	1	-	-	-	1	0	0
F_3	-	1	-	1	-	0	0
F_4	-	1	-	-	1	0	0
F_5	-	-	1	1	-	0	0
F_6	-	-	1	-	1	0	0

sublist containing the index terms which belong to the document. This type of organization results in a sequentially structured file (SSF). (Sometimes this type of file is called a linear file.)

To implement the decision function on an SSF, the master list of document numbers is examined sequentially. The sublist of index terms associated with each document number is scanned to determine if any of the 'feature terms' are present. If so, their weights are summed and the result compared to the threshold. All relevant documents in the file can be identified by repeating this operation for each document number in the master list. It is also possible to see if index term combinations in each document sublist match those specified by each template in the BRS. Thus for an SSF the relevant documents can be recognized by summing the term weights directly, <u>or</u> by using the template matching technique with a BRS.

The major disadvantage of an SSF is that all records in the file must be individually inspected to identify a very small subset of relevant documents. The cost of searching an SSF increases proportionally with the number of document records it contains.

To reduce the unit cost of identifying relevant documents in a file, the file can be organized in a different manner. Here the master list is composed of the individual index terms in some order. Each index term in the master list has an associated sublist of document numbers. Each document numbered in the sublist is indexed with the term in the master list. This type of file can be called an inversely structured file (ISF).

To implement the decision function on an ISF the matching templates of the BRS are necessary. Index term weights cannot be applied. The individual BRS templates are matched by set intersection operations on all index terms corresponding to fixed indices in the solution families. The set operations are performed only on the sets of document numbers which are associated with index terms which are 'features'. These feature sets are a small fraction of the total file. Thus the unit costs of recognizing patterns (relevant documents) are lower in an ISF than in an SSF. However, the increased search efficiency is offset in part by the extra costs incurred by organizing the ISF. (The natural ordering is the SSF.)

3.34 Example Showing System Operation

Figures 3-2 and 3-3 illustrate how the decision function is derived and how the documents predicted to be relevant are identified using both a direct weighted term approach and the BRS templates.

Figure 3-2A shows the matrix model which might arise from the selection of five index terms as features. The training set contains eight documents with $y_i = +1$ for relevant documents and $y_i = -1$ for nonrelevant documents. The relevance threshold τ for this model is taken to be zero. The best approximate solution (in the L_1 sense) is shown in Fig. 3-2B. This also shows the residual vector $\underline{r}$ with $L_1(\underline{r}) = \sum |r_i| = 3$.

Figure 3-2C shows the decision function, or linear utility prediction equation (LUPF). When this function is thresholded (using

$\tau=0$) a linear pseudo-Boolean inequality (LPBI) results which has six solution families as shown.

Figure 3-3 shows all 32 possible combinations of the five index terms which were extracted as features. The predicted utility of each combination is shown as it would be determined by a direct summing of the index term weights. This approach might be taken with an SSF.

The groups of combinations with $\hat{u} \geq 0$ which are specified by the solution families (templates) of the BRS are identified for comparison. This approach to identifying relevant documents would be taken with an ISF.

FIGURE 3-2

SAMPLE PROBLEM ILLUSTRATING DERIVATION OF THE DECISION FUNCTION AND BRS

A. Matrix Model Arising from Training Set of Documents

$\underline{y} = Z\underline{\beta} + \underline{r}$

$$\begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ +1 \\ +1 \\ -1 \\ +1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \beta_o \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix} + \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{bmatrix}$$

B. Best Approximate L_1 Solution $\underline{b}$ and Residual Vector $\underline{r}$

$$\underline{b} = \begin{bmatrix} b_o \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 1 \\ -1 \end{bmatrix} ; \underline{r} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \\ 0 \end{bmatrix} ; L_1(\underline{r}) = \sum_{i=1}^{8} |r_i| = 3$$

C. LUPF, LPBI AND BRS

LUPF: $\hat{u} = T_1 - T_2 + T_4 - T_5$

LPBI: $\hat{u} \geq \tau = 0 \Rightarrow T_1 - T_2 + T_4 - T_5 \geq 0$

BRS:

$$\boxed{\begin{aligned} F_1(\underline{T}) &= (1,0,-,-,-) \\ F_2(\underline{T}) &= (1,1,-,1,-) \\ F_3(\underline{T}) &= (1,1,-,0,0) \\ F_4(\underline{T}) &= (0,0,-,1,-) \\ F_5(\underline{T}) &= (0,0,-,0,0) \\ F_6(\underline{T}) &= (0,1,-,1,0) \end{aligned}}$$

FIGURE 3-3

PREDICTED UTILITIES FOR COMBINATIONS OF INDEX TERMS

Combination number	Index term configuration					Predicted utility $\doteq \hat{u}$		
	T_1	T_2	T_3	T_4	T_5			
1	1	1	1	1	1	0		
2	1	1	1	1	0	1	F_2	
3	1	1	1	0	1	-1		
4	1	1	1	0	0	0	F_3	
5	1	1	0	1	1	0		
6	1	1	0	1	0	1	F_2	
7	1	1	0	0	1	-1		
8	1	1	0	0	0	0	F_3	
9	1	0	1	1	1	1		
10	1	0	1	1	0	1		
11	1	0	1	0	1	0		
12	1	0	1	0	0	1	F_1	Solution families
13	1	0	0	1	1	1		
14	1	0	0	1	0	2		
15	1	0	0	0	1	0		
16	1	0	0	0	0	1		
17	0	1	1	1	1	-1		
18	0	1	1	1	0	0	F_6	
19	0	1	1	0	1	-2		
20	0	1	1	0	0	-1		
21	0	1	0	1	1	-1		
22	0	1	0	1	0	0	F_6	
23	0	1	0	0	1	-2		
24	0	1	0	0	0	-1		
25	0	0	1	1	1	0		
26	0	0	1	1	0	1	F_4	
27	0	0	1	0	1	-1		
28	0	0	1	0	0	0	F_5	
29	0	0	0	1	1	0		
30	0	0	0	1	0	1	F_4	
31	0	0	0	0	1	-1		
32	0	0	0	0	0	0	F_5	

4.0 AN INFORMATION THEORETIC MEASURE FOR RANKING AND SELECTING INDEX TERMS

4.1 Introduction

An information theoretic measure of goodness is developed for ranking index terms found in a training set of documents. Each index term is regarded independently as a potential 'experiment' which can be used to predict the relevance of documents in the training set.

For example, knowing that there are 20 relevant and 30 non-relevant documents in a training set, but lacking any other information, a decision maker if presented with a document selected at random from the training set, would assume that the probability of the document being relevant (before he examines it) is 0.40. Suppose now that before inspecting the document and making his decision about relevance, the user is shown one index term associated with the document. If he knows that this term occurred with 20 of the training set documents and that 15 of these 20 were relevant, then the user would be justified in concluding that the probability of the document being relevant is 0.75.

Knowledge that the particular index term was present has provided <u>information</u> (or resolved uncertainty) about the classification of the document. In fact it will provide (on the average and for this example using the above data) 0.18 <u>bits</u> of information each time it is found with a document. The development of this quantitative measure of information (divorced from economic considerations) will be presented here. This measure is used to select the best index terms, i.e., those terms which individually provide the most information about document relevance.

4.2 The Decision Theory Model

A simple decision theory model is shown below (see Hadley[60], or Fishburn[61] for a more thorough discussion).

$$
\begin{array}{c|cccc|}
 & p(x_1) & p(x_2) & & p(x_n) \\
 & x_1 & x_2 & \cdots & x_n \\
\hline
a_1 & u_{11} & u_{12} & \cdots & u_{1n} \\
a_2 & u_{21} & u_{22} & \cdots & u_{2n} \\
\vdots & \vdots & \vdots & & \vdots \\
a_r & u_{r1} & u_{r2} & \cdots & u_{rn} \\
\hline
\end{array}
$$

There are n 'states of nature' or possible outcomes x_j, $j=1,2,\ldots,N$ which are relevant to the decision maker's problem. The probability distribution $p(X) = \{p(x_1),\ldots,p(x_n)\}$ over these states of nature is assumed known to the decision maker. A random experiment is performed which determines which state of nature x_j actually holds. The results of this experiment are <u>not</u> available to the decision maker.

The decision maker has a set of r possible actions a_i, $i=1,2,\ldots,r$ which he can take. One and only one of the actions a_i must be selected.

After the action has been selected by the decision maker, the true state of nature x_j is revealed to him. He will then receive the reward u_{ij}, which may be negative. (u_{ij} is a utility, which includes monetary as well as more subjective rewards.)

The decision problem is solved when the decision maker chooses an action. The best action a_ℓ is one which maximizes the expected utility; i.e.

$$a_\ell = \max_{i \leq r} \left(\sum_j u_{ij} p(x_j) \right) .$$

4.21 Decision Problems with Experimentation

A natural extension of the decision theory model discussed above is to allow the decision maker to perform an auxiliary experiment before picking an action[62]. Recall that the state of nature x_j has already been determined, but the results are unknown to him. This experiment can be considered to be an attempt to gain more information about the true state of nature.

Define $Y = \{y_1, y_2, \ldots, y_S\}$ as the event set for the experiment performed by the decision maker, i.e., these are the only outcomes.

It is assumed that the conditional distributions

$$p(Y/x_j) = \{p(y_1/x_j), \ldots, p(y_S/x_j)\}, \; j=1,2,\ldots,n$$

are known to the decision maker, as well as $p(X) = \{p(x_1), \ldots, p(x_n)\}$

4.211 Bayes Rule. It is a trivial consequence of the definition of conditional probabilities

$$p(y_k/x_j) = \frac{p(y_k,x_j)}{p(x_j)}$$

that we are able to write

$$p(x_j/y_k) = \frac{p(x_j,y_k)}{p(y_k)} .$$

Thus

$$p(x_j)p(y_k/x_j) = p(x_j/y_k)p(y_k)$$

$$\Longrightarrow \quad p(x_j/y_k) = \frac{p(x_j)}{p(y_k)}\, p(y_k/x_j) .$$

Now using

$$p(y_k) = \sum_j p(y_k/x_j)p(x_j) = \sum_j p(y_k,x_j) ,$$

we have

$$p(x_j/y_k) = \frac{p(x_j)p(y_k/x_j)}{\sum_j p(y_k/x_j)p(x_j)} .$$

This last expression is known as Bayes Rule[63]. $p(X/y_k) = \{p(x_j/y_k), k=1,\ldots,s\}$ is a <u>new</u> probability distribution over the n states of nature.

The interpretation here is that for any particular observed experimental outcome y_k, an entire new probability distribution $p(X/y_k)$ may be constructed. Since the experiment has S possible outcomes, there are S possible new distributions which may be derived.

To distinguish between the initial distribution $p(X)$ and the distributions $p(X/y_k)$ derivable after the experimental outcome y_k has been observed, it has become customary to call $p(X)$ the <u>prior distribution</u> and $p(X/y_k)$ the <u>posterior distribution</u>.

To perform the transformation from prior to posterior distributions, it is necessary to know <u>both</u> the prior distribution $p(X)$ and the conditional distributions $p(Y/x_j)$, $j=1,2,\ldots,n$. This knowledge is <u>equivalent</u> to knowing the <u>joint distribution</u>

$$p(y_k,x_j) = p(y_k/x_j)p(x_j),\ j=1,2,\ldots,n,\ k=1,2,\ldots,S.$$

After the posterior distribution $p(X/y_k)$ is determined, it is used in place of the prior distribution to determine the action $a_{\ell(k)}$ having the maximum expected utility, i.e.

$$a_{\ell(k)} = \max_{i\leq r}\left(\sum_j p(x_j/y_k)u_{ij}\right).$$

The experiment has allowed a better, more up-to-date estimation of the state of nature.

4.3 Selection of Experiments

The purpose of the experiment performed by the decision maker is to provide more information about the true state of nature. The information is conveyed by permitting a revision of the probability distribution over the state of nature from $p(X)$ to $p(X/y_k)$.

In many problems, the decision maker can choose from a group of experiments only one which will be performed to obtain $p(X/y_k)$. This raises the interesting question of which experiment is 'best'. That is, how can experiment 'goodness' be defined to permit a ranking of all available experiments?

4.31 Decision Theory Approach when the Utilities are Known

In the context of the decision model discussed above, when the utilities u_{ij} are known, the answer is to pick the experiment which maximizes the expected utility averaged over all possible posterior distributions.

For each experiment, consider each outcome y_k in turn and using the associated posterior distribution $p(X/y_k)$ determine the maximum utility which will result from making the best decision, using this distribution. Then weight these utilities by the marginal probabilities $p(y_k)$ that the outcomes will occur. This gives the expected utility for each experiment assuming the best decision is always made for each possible outcome. Finally, the 'best' experiment is the one with the highest average utility (averaged over all possible posterior distributions).

4.32 Inadequacy of the Decision Theory Model when the Utilities are Not Known

There are at least three situations which frequently arise and make the above procedures inapplicable.

(A) The utilities are all equal. In this case the expected costs of all actions are equal and a best action cannot be chosen.

(B) The utilities are unknown, or fluctuate to such an extent that they can be considered to be unknown.

(C) The utilities do not exist, but a prior distribution can be postulated; and various observed variables can give rise to posterior distributions.

Situation (B) above might occur for example, where a local decision problem exists within a large system. The global utility of selecting various local experiments is not estimable in this case. Such types of situations are felt to arise frequently in design problems, where small portions of the overall system are designed independently of the others.

Situation (C) arises most often from a purely analytical situation where no utilities are associated with a choice of experiment.

All three of the above situations negate the selection of information-gathering experiments by using an expected utility measure.

However, the fact that experiments do provide information remains, whether or not an economic value can be attached to the information.

The process of index term selection can be modeled in the context of a prior distribution which is modified by experimental information to give posterior distributions. However, utilities are not easily defined.

For the evaluation of these processes without attaching an economic measure, we turn now to information theory.

4.4 Results from Information Theory[1]

4.41 Definition of Entropy

As a definition, let

$$H(P) = H(p_1,\ldots,p_n) = -C \sum_{i=1}^{n} p_i \ln p_i$$

be called the entropy of the probability distribution

$$P = \{p_1,p_2,\ldots,p_n\}; \text{ where } \sum_{i=1}^{n} p_i = 1;\ p_i \geq 0 .$$

The functional form of $H(P)$ is determined up to a multiplicative constant by specifying the three conditions given below.

[1]Analytical developments presented here closely follow those presented by A. Feinstein(64). As a secondary source, see S. Watanabe(65).

(A) $H(p, 1-p)$ is a continuous function of p for $0 \leq p \leq 1$.

(B) $H(P)$ is a symmetric function of all its variables.

(C) If $p_n = q_1 + q_2 > 0$, then

$$H(p_1, \cdots, p_{n-1}, q_1, q_2) = H(p_1, p_2, \cdots, p_n) + p_n H\left[\frac{q_1}{p_n}, \frac{q_2}{p_n}\right].$$

By agreeing to take logarithms to the base 2 and by setting C=1, the units of information become bits. We shall denote this by writing

$$H(P)' = -\sum_{i=1}^{n} p_i \log p_i$$

with the understanding that $0 \log 0 = 0$.

It is possible to prove the following two important results(66) given below.

(A) The entropy $H(P)$ is bounded. That is, $0 \leq H(P) \leq \log n$ with $H(P) = 0$ iff $p_k = 1$ for some k, and $H(P) = \log n$ iff $p_j = 1/n$ for all j.

(B) $H(P)$ is strictly concave[1].

Result (A) has an intuitive interpretation when the entropy is regarded as the uncertainty in the probability distribution P.

---

[1]This follows from the fact that $z = -p \log p$ is strictly concave:

$$\frac{d^2z}{dp^2} < 0 \quad \text{for} \quad p \geq 0.$$

Let $p_k = p(x_k) = 1$; $p_j = 0$, $j \neq k$. In this case, event x_k is a certainty, and the entropy is zero. Let $p_j = 1/n$; $j=1,2,\ldots,n$. In this case all events x_j are equally uncertain and the entropy is a maximum.

By result (B), the function $H(P)$ smoothly approaches its single maximum value. Intuitively, this allows us to rank all probability distributions without ambiguity according to their entropy, in the sense that distributions with greater entropy are always closer to the maximum entropy distribution given by $p_j = 1/n$.

Figure 4-1 shows the entropy for the two state distribution $p_1 + p_2 = 1$; $p_1, p_2 \geq 0$. The maximum entropy of one bit is attained when $p_1 = p_2 = 1/2$. The maximum is fairly broad.

4.42 Definitions of Event Sets and Probabilities

Let $X = \{x_1,\ldots,x_n\}$ and $Y = \{y_1, y_2, \ldots, y_m\}$ be two finite discrete sets of events. Denote by $X \bigotimes Y$ the product set consisting of all mn pairs (x_1, y_j).

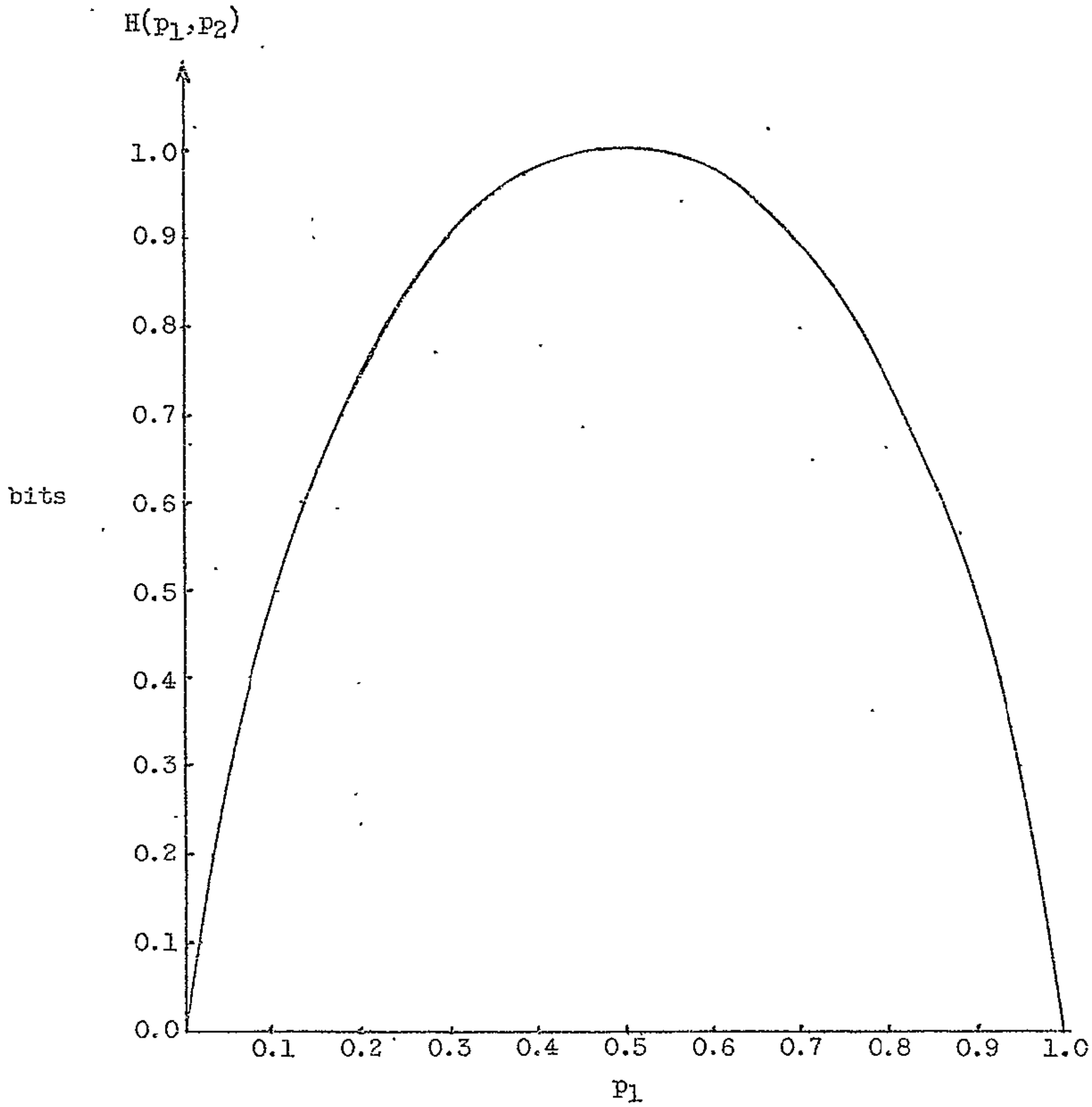
Assume that there is a probability distribution defined over $X \bigotimes Y$ with probabilities denoted by $p(x_1, y_j)$. This is the joint distribution of X and Y, $p(X,Y)$, where

$$p(x_i, y_j) \geq 0; \; i=1,2,\ldots,n; \; j=1,2,\ldots,m$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) = 1.$$

FIGURE 4-1

ENTROPY PLOT OF A SIMPLE BINARY DISTRIBUTION
AS A FUNCTION OF ONE PROBABILITY



where $H(p_1,p_2) = -[p_1 \log_2 p_1 + p_2 \log_2 p_2]$
and $p_1 + p_2 = 1$; $p_1, p_2 \geq 0$

Let the marginal probabilities be given by

$$p(x_i) = \sum_{j=1}^{m} p(x_i, y_j), \; i=1,2,\cdots,n;$$

and

$$p(y_j) = \sum_{i=1}^{n} p(x_i, y_j), \; j=1,2,\cdots,m.$$

Then denote the marginal distributions by $p(X)$ and $p(Y)$.

Define conditional probabilities as

$$p(x_i/y_j) = \frac{p(x_i, y_j)}{p(y_j)} \; ; \; p(y_j) > 0 \; .$$

and

$$p(y_j/x_i) = \frac{p(x_i, y_j)}{p(x_i)} \; ; \; p(x_i) > 0 \; .$$

Then let the conditional distributions be given by

$$p(X/y_j), \; j=1,2,\cdots,m$$

and

$$p(Y/x_i), \; i=1,2,\cdots,n.$$

4.43 Entropy of the Distributions

It is useful to define the entropies of the joint distributions, the marginal distributions and the conditional distributions as shown below.

(A) $$H(X,Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i,y_j) \log p(x_i,y_j)$$ is the entropy of the joint distribution.

(B) The entropies of the marginal distributions are given by

$$H(X) = -\sum_{i} p(x_i) \log p(x_i)$$

and

$$H(Y) = -\sum_{j} p(y_j) \log p(y_j).$$

(C) Define the entropy of each conditional distribution as

$$H(X/y_j) = \sum_{i=1}^{n} p(x_i/y_j) \log p(x_i/y_j);\ j=1,2,\cdots,m.$$

Then the <u>average entropy</u> of all conditional distributions is defined by

$$H(X/Y) = \sum_{j=1}^{m} p(y_j)H(X)/y_j)$$

$$= \sum_{j=1}^{m} p(y_j) \sum_{i=1}^{n} p(x_i/y_j) \log p(x_i/y_j)$$

$$= \sum_{j}^{m} \sum_{i}^{n} p(x_i, y_j) \log p(x_i/y_j).$$

4.44 Useful Relationships between Entropies of Distributions

The relations shown below for distributional entropies can be proven by using the previous definitions:

$$H(X,Y) = H(Y) + H(X/Y) = H(X) + H(Y/X) \tag{4-1}$$

$$H(X,Y) \leq H(X) + H(Y) \tag{4-2}$$

with equality iff $p(X)$ and $p(Y)$ are statistically independent.

$$0 \leq H(X/Y) \leq H(X) \tag{4-3}$$

$$R = H(X) - H(X/Y) = H(Y) - H(Y/X) \geq 0 \tag{4-4}$$

$$R = H(X) + H(Y) - H(X,Y) \tag{4-5}$$

4.5 Interpretation of Information Theoretic Results

4.51 Bayesian Interpretation

The above results are all we need to describe information in quantitative, non-economic terms.

Intuitively, the entropy of a distribution represents the uncertainty in the distribution. If we revise the distribution from prior to posterior through Bayes rule after observing the results of an experiment, how does the entropy change?

By letting $H(X)$ be identified with the <u>uncertainty</u> in the prior distribution, it follows that $H(X/y_j)$ is the <u>uncertainty</u> in the posterior distribution obtained from Bayes rule after observing one particular experimental outcome y_j; $j=1,2,\ldots,m$. Since there are m possible posterior distributions, it is reasonable to define $H(X/Y)$ as the <u>average uncertainty</u> over all posterior distributions.

It is customary and intuitively pleasing to define a decrease in uncertainty (entropy) as in increase in <u>information</u>, or $I = \Delta H = H_1 - H_0$. This allows the amount of information gathered to be measured in bits. In this sense then, $R = H(X) - H(X/Y)$ is the measure of <u>information provided by the experiment</u>. From (4-4), this information will always be positive. <u>Each time</u> the experiment is performed R bits of information (on the average) are acquired. If the experiment is very good, $H(X/Y) = 0$ and the posterior distribution has no uncertainty. Here $R = H(X)$ and all the uncertainty in the prior distribution has been removed by the experiment. If the experiment is very poor, then $H(X/Y) = H(X)$ and no information has been provided by the experiment. In this case $R = 0$.

Of course the amount of information which can be provided by an experiment is limited by the amount of uncertainty contained in the prior distribution. Thus for a given prior distribution, the best experiment is the one with the largest value of R. To compare experiments in decision problems with different prior distributions it is convenient to define a dimensionless figure of merit

$$\alpha = \frac{R}{H(X)}$$

where $0 \leq \alpha \leq 1$. $PCT = 100\alpha$ is the percent of uncertainty in the prior distribution which is resolved by the experiment. $PCT = 100$ implies a perfect experiment and $PCT = 0$ implies a worthless experiment.

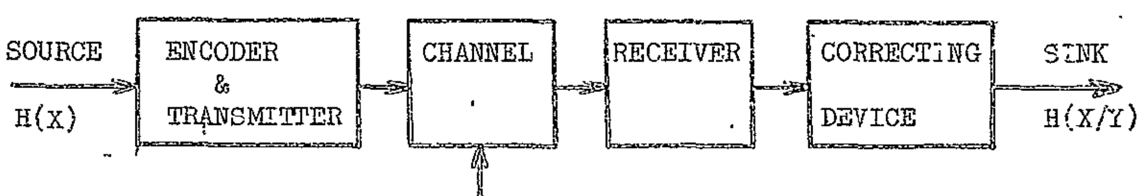
Relation (4-4) states that the goodness of an experiment can also be measured by $R = H(Y) - H(Y/X)$. Here $H(Y)$ is a function of the experiment alone. $H(Y/X)$ is the average uncertainty in Y, if X is known beforehand. $R = H(Y) - H(Y/X)$ is the amount of information about Y which is acquired from knowing X. This expresses an information balance[67]. The amount of information contained about X in Y is equal to the amount of information about Y in X.

From (4-4) it is clear then that the goodness of an experiment can be inferred from either the average amount of information provided by the experiment as to the state of nature, or the average amount of information provided by the state of nature as to the outcome of the experiment. This is simply the strength of the statistical dependence between cause and effect, or effect and cause. From (4-5) and (4-2), if cause and effect are statistically independent, R = 0.

The interpretation of cause and effect relationships is discussed in depth by Watanabe[68]. His conclusions regarding interpretation of entropy expressions are similar to those presented here. He defines the inferential process of looking ahead from a known state of nature to the uncertain outcome of an experiment as being <u>prediction</u>, and looking backward from a known experimental outcome to the uncertain state of nature as being <u>retrodiction</u>.

4.52 Communication Theory Interpretation

The decision theory interpretation of entropy reduction by performing an experiment is not the customary way to interpret relations (4-1) through (4-5). Communication engineers prefer to interpret the same results in terms of an information (or symbol) transmitter, a noisy channel, and a receiver, as shown below[69].



Here discrete symbols are drawn randomly from a probability distribution p(X) having entropy H(X), and are transmitted sequentially (as drawn) through a noisy channel. A distorted message is received, where distortion implies that some of the symbols are changed by the noise into different symbols. A correcting device attempts to infer what symbol was sent, on the basis of what symbol is received. $H(X/Y)$ is the residual entropy associated with the message received after the correcting device has 'cleaned up' the noisy message. $H(X/Y)$ is referred to as the <u>equivocation</u> of the channel with respect to the source distribution $p(X)$. It represents the amount of information <u>lost</u> (not recoverable by the correcting device) in the channel. $R = H(X) - H(X/Y)$ is the amount of information transmitted through the noisy channel.

Both the decision theory and the communications theory interpretation of information theoretic expressions have merit, depending on the problem at hand.

4.53 Computation of an Information Statistic $\hat{R}$

For computational purposes, consider a decision problem with two states of nature, and an associated experiment with two outcomes. After observing the true states of nature and the corresponding experimental outcome for several trials, it is possible to summarize the observations in the sample contingency table of integers shown below.

Outcomes of experiment

		y_1	y_2	
states of nature	x_1	n_{11}	n_{12}	$n_{11} + n_{12} = n_{1\cdot}$
	x_2	n_{21}	n_{22}	$n_{21} + n_{22} = n_{2\cdot}$
		$n_{11} + n_{21} = n_{\cdot 1}$	$n_{12} + n_{22} = n_{\cdot 2}$	$N = n_{11} + n_{12} + n_{21} + n_{22} = n_{\cdot\cdot}$

(4-6)

There is a large body of literature which deals with the statistical theory of contingency tables. See for example Kullback[70]. However (4-6) above will be considered here simply as a convenient tabular data array. Data in (4-6) will be used to estimate R.

Let $\hat{R}$ be a sample estimate of R based on the observations in (4-6). $\hat{R}$ will henceforth be called the information statistic. It can be computed directly from either (4-4) or (4-5). However, it is easy to derive a more convenient computational form. To do this, first define a contingency table of probability estimates (the joint distribution $p(X,Y)$ as follows:

	y_1	y_2	
x_1	α	β	$\alpha + \beta$
x_2	γ	δ	$+ \delta$
	$\alpha + \gamma$	$\beta + \delta$	1.0

$$\alpha = n_{11}/N$$
$$\beta = n_{12}/N$$
$$\gamma = n_{21}/N$$
$$\delta = n_{22}/N \qquad (4\text{-}7)$$

Then:

$$\hat{R} = \hat{H}(X) + \hat{H}(Y) - \hat{H}(X,Y) \tag{4-5}$$

$$= -(\alpha + \beta)\log(\alpha + \beta) - (\gamma + \delta)\log(\gamma + \delta) - (\alpha + \gamma)\log(\alpha + \gamma)$$

$$-(\beta + \delta)\log(\beta + \delta) + \alpha\log\alpha + \beta\log\beta + \gamma\log\gamma + \delta\log\delta.$$

Collecting all terms in α, β, γ, and δ gives:

$$\hat{R} = \alpha[-\log(\alpha + \beta) - \log(\alpha + \gamma) + \log\alpha]$$

$$+ \beta[-\log(\alpha + \beta) - \log(\beta + \delta) + \log\beta] + \gamma[-\log(\alpha + \gamma) - \log(\gamma + \delta)$$

$$+ \log\gamma] + \delta[-\log(\beta + \delta) - \log(\gamma + \delta) + \log\delta]$$

$$= \alpha\log\left[\frac{\alpha}{(\alpha + \beta)(\alpha + \gamma)}\right] + \beta\log\left[\frac{\beta}{(\alpha + \beta)(\beta + \delta)}\right]$$

$$+ \gamma\log\left[\frac{\gamma}{(\alpha + \gamma)(\gamma + \delta)}\right] + \delta\log\left[\frac{\delta}{(\beta + \delta)(\gamma + \delta)}\right];$$

or, in terms of the integer counts

$$N\hat{R} = n_{11}\log\left[\frac{Nn_{11}}{(n_{11} + n_{12})(n_{11} + n_{21})}\right] + n_{12}\log\left[\frac{Nn_{12}}{(n_{11} + n_{12})(n_{12} + n_{22})}\right]$$

$$+ n_{21}\log\left[\frac{Nn_{21}}{(n_{21} + n_{22})(n_{21} + n_{11})}\right] + n_{22}\log\left[\frac{Nn_{22}}{(n_{21} + n_{22})(n_{12} + n_{22})}\right].$$

Since $n_{.j} = \sum_{i=1}^{2} n_{ij}$ and $n_{i.} = \sum_{j=1}^{2} n_{ij}$

we get: $$N\hat{R} = \sum_{i=1}^{2}\sum_{j=1}^{2} n_{ij}\log\left[\frac{Nn_{ij}}{(n_{i.})(n_{.j})}\right]. \tag{4-8}$$

This gives a convenient computational form for the information statistic $\hat{R}$. However when $\hat{\alpha} = \hat{R}/\hat{H}(X)$ is to be computed, direct use of (4-4) is recommended, since $\hat{H}(X)$ is produced as a byproduct.

If $\hat{R}$ is the estimated number of bits of information (on the average) which are provided each time the experiment is performed, then $N\hat{R}$ is the total number of bits of information provided by all the N replications of the experiment.

There is another interpretation of the information statistic based on (4-8).. Suppose the sample contingency table arises from comparing a (0/1) vector $\underline{x}$ (two states of nature, zero and one) with a (0/1) experimental outcome vector $\underline{y}$ (two experimental outcomes, zero and one). The similarity of vectors $\underline{x}$ and $\underline{y}$ is intuitively high if $x_i = y_i = 0$ or 1 for a large number of indices i. Of the four terms in the expression (4-8), two involve n_{ij} on the main diagonal of the table, and two involve n_{ij} off the diagonal. The sum of the diagonal terms of (4-8) represents the measure of <u>similarity</u> between the vectors $\underline{x}$ and $\underline{y}$, while the sum of the off-diagonal terms is a measure of their <u>dissimilarity</u>.

4.54 Statistical Distribution of the Information Statistic

Since $\hat{R}$ is a statistic drawn from a sample, it can be expected to behave as a random variable. It is known that[71]

$$[\log_e 2]2N\hat{R}$$

is asymptotically distributed as a central chi-squared variable with one degree of freedom (for a 2 x 2 sample contingency table) under the null hypothesis that $R = 0$. The factor $\log_e 2 = 0.693$ is needed because $\hat{R}$ is assumed to have the units of bits in (4-8).

4.55 Example Problem

As an example, consider a training set of 28 documents. A set of 155 index terms were found with this document set. An estimate of the information provided about document relevance by two of these terms will be made to illustrate previous results.

Vector $\underline{x} = (x_i)$, $i=1,2,\cdots,28$, of Fig. 4-2A shows the correct classification of each of the 28 documents in the training set, with $x_i = 1$ if document i is relevant. Vectors $\underline{T}_1 = (t_{i1})$ and $\underline{T}_2 = (t_{i2})$ of Fig. 4-2A show how terms 1 and 2 are used to index the 28 documents. For example, if $T_{i1} = 1$, then index term 1 is used to index document i.

It is possible to compare the effectiveness of terms 1 and 2 as relevance indicators (over the training set) by comparing vectors $\underline{T}_1$ and $\underline{T}_2$ separately with vector $\underline{x}$. Fig. 4-2B shows the results of these comparisons expressed as 2 x 2 contingency tables. Calculations leading to α_1 and α_2 are detailed in Fig. 4-2C. Equation (4-4) is used for $\hat{R}$ instead of (4-8) because $\hat{H}(X)$ is generated as a byproduct with (4-4), and $\hat{H}(X)$ is required for $\hat{\alpha} = \hat{R}/\hat{H}(X)$. Fig. 4-2C shows the estimated marginal and conditional distributions and their corresponding entropies. It can be seen that term 2 ($\hat{\alpha}_2 = 0.0780$) is estimated to be slightly better than term 1 ($\hat{\alpha}_1 = 0.0701$).

FIGURE 4-2

EXAMPLES ILLUSTRATING COMPUTATION OF AN INFORMATION STATISTIC FOR ESTIMATING INFORMATION ABOUT DOCUMENT RELEVANCE CONVEYED BY INDEX TERMS.

A. Vectors for Comparison

i	X_i	$t_{i,1}$	$t_{i,2}$
1	1	0	0
2	0	1	1
3	1	0	0
4	1	1	0
5	0	0	0
6	1	0	0
7	0	0	0
8	0	1	0
9	0	0	1
10	1	0	0
11	0	0	0
12	1	0	0
13	1	0	0
14	0	0	0
15	0	0	0
16	0	1	0
17	0	0	0
18	0	0	1
19	0	1	0
20	0	0	0
21	0	1	0
22	1	1	0
23	1	0	0
24	0	1	0
25	1	0	0
26	0	1	0
27	0	1	0
28	0	1	0

B. Contingency Tables for Comparing $\underline{X}$ with $\underline{T}_1$ and $\underline{T}_2$

	$t_{i,1}=0$	$t_{i,1}=1$	
$X_i=0$	9	9	18
$X_i=1$	8	2	10
	17	11	28

	$t_{i,2}=0$	$t_{i,2}=1$	
$X_i=0$	15	3	18
$X_i=1$	10	0	10
	25	3	28

C. Computations*

	$\underline{x}$ with $\underline{T}_1$	$\underline{x}$ with $\underline{T}_2$
$p(X)$	(0.64286, 0.35714)	(0.64286, 0.35714)
$H(X)$	0.94027	0.94027
$p(X/t_i=0)$	(0.52942, 0.47058)	(0.600, 0.400)
$H(X/t_i=0)$	0.99749	0.97096
$p(X/t_i=1)$	(0.81818, 0.18182)	(1.00, 0.00)
$H(X/t_i=1)$	0.68402	0.00
$p(T)$	(0.60714, 0.39286)	(0.89286, 0.10714)
$H(X/T)$	0.8743	0.86694
$\hat{R}=H(X)-\hat{H}(X/T)$	0.06593	0.07333
$\alpha=\hat{R}/H(X)$	0.0701	0.0780

*$p(\cdot)$ is the probability distribution and $H(\cdot)$ is the distribution entropy

5.0 SOLVING THE DISCRETE LINEAR APPROXIMATION PROBLEM IN THE L_1 NORM

5.1 Introduction

The discrete linear approximation model can be written as follows

$$\underline{y} = X\underline{\beta} + \underline{r}.$$

The model can also be written as

$$y_i = \beta_o + \sum_{j=1}^{n-1} \beta_j \phi_{ij} = \sum_{j=0}^{n-1} \beta_j \phi_{ij}; \ i=1,2,\cdots,m.$$

The linear approximation problem arises when <u>estimates</u> of the unknown vector $\underline{\beta}$ are desired. We define a <u>best estimate</u> of $\underline{\beta}$ to be the vector $\underline{b}^*$ which minimizes the length of the residual vector $\underline{r}$. If we designate the length of the vector $\underline{r}$ by $||\underline{r}||$, called the norm of $\underline{r}$, then our approximation problem becomes:

Find $\underline{b}^*$ such that

$$\underline{b}^* = \min_{\underline{b}} ||\underline{r}|| = \min_{\underline{b}} ||\underline{y} - X\underline{b}|| .$$

A class of norms is given by[72,73]

$$||\underline{r}|| = \left(\sum_i |r_i|^p\right)^{1/p} \quad \text{for } 1 \leq p \leq \infty .$$

When $p = 2$, the familiar least squares problem results. The cases where $p = 1$ and $p = \infty$ are also of practical interest because algorithms are available to compute $\underline{b}^*$. In particular, they may be formulated as linear programming problems and be easily solved.

$L_1(\underline{r})$ corresponding to $p = 1$ gives a fit which minimizes the sum of the absolute values of the residuals r_i, $i=1,2,\cdots,n$. $L_\infty(\underline{r})$ corresponding to the limiting case $L_\infty(\underline{r}) = \lim_{p\to\infty} L_p(\underline{r}) = \max_{1\le i\le n} |r_i|$ gives a fit which minimizes the largest residual (in absolute value). The L_∞ norm is also often called the <u>uniform</u> or <u>Chebyshev</u> norm.

The L_1 and L_∞ solutions will always exist when computed using the linear programming formulation, even when the rank of X is $q < n$. This makes the L_1 and L_∞ norms attractive when dealing with data matrices which are not known beforehand to have rank $q = n$. The L_2 (least squares solution) normal equations do not have a solution when $q < n$.

For the application considered here, the approximation problem arises when index term 'weights' are to be derived for estimating document utility. The matrix X is not known beforehand to have rank $q = n$. The L_1 norm is used here to estimate the index term weights, and no problem is encountered if $q < n$. In addition the solution is very rapidly and conveniently attained with the linear programming formulation. Formulation of the L_1 problem as a linear program is briefly reviewed below. Example problems are used to illustrate the development.

5.2 Formulating the Discrete L_1 Problem as a Linear Programming Problem

Formulation of the L_1 problem as a linear programming problem has been shown by I. Barrodale[74,75] and P. Rabinowitz[76]. The formulation proceeds as follows: let

$$\underline{y} = X\underline{\beta} + \underline{r}.$$

Now, since $\underline{\beta}$ and $\underline{r}$ are unrestricted in sign, they can each be expressed as the difference between two non-negative vectors, i.e.

$$\underline{\beta} = \underline{\beta}^+ - \underline{\beta}^-;\ \underline{\beta}^+,\underline{\beta}^- \geq \underline{o}$$

$$\underline{r} = \underline{r}^+ - \underline{r}^-;\ \underline{r}^+,\underline{r}^- \geq \underline{o}$$

$$\Longrightarrow\quad \underline{y} = X(\underline{\beta}^+ - \underline{\beta}^-) + (\underline{r}^+ - \underline{r}^-)$$

$$\Longrightarrow\quad (X|-X|I|-I)\begin{pmatrix}\underline{\beta}^+\\ \underline{\beta}^-\\ \underline{r}^+\\ \underline{r}^-\end{pmatrix} = \underline{y}.$$

These equations can be regarded as the <u>constraint set</u> for a linear programming problem. The unknowns are the vectors $\underline{\beta}^+,\underline{\beta}^-,\underline{r}^+,\underline{r}^-$. The distinction made in section 5.1 between the unknown vector $\underline{\beta}$ and its optimal estimate $\underline{b}^*$ has been dropped here to eliminate notational complexity. All vectors $\underline{\beta}$ appearing as the unknowns in LP problems are to be considered <u>estimates</u> of the true vectors.

The objective function can be formulated by observing that unit vectors corresponding to r_i^+ and r_i^- will never be in the basis at the same time, since they are linearly dependent, (the same remarks apply to $\underline{\beta}_i^+$ and $\underline{\beta}_i^-$, see Hadley[77]). The solution variable $r_i^+ + r_i^-$ then represents the absolute value of the ith residual, since:

$$\text{either } r_i^+ = |r_i| \geq 0 \text{ and } r_i^- = 0;$$
$$\text{or } r_i^- = |r_i| \geq 0 \text{ and } r_i^+ = 0.$$

By putting zero costs in for the unknowns β_i^+ and β_i^- and unit costs in for the unknowns r_i^+ and r_i^-, the sum of the absolute values of the residuals is minimized. This gives the linear programming problem shown below.

$$\text{Minimize} \qquad z = \sum_{i=1}^{n} 0.\beta_i^+ + \sum_{i=1}^{n} 0.\beta_i^- + \sum_{i=1}^{m} 1.r_i^+ + \sum_{i=1}^{m} 1.r_i^-$$

$$\text{subject to} \qquad (X|-X|I|-I)\begin{pmatrix}\underline{\beta}^+\\ \underline{\beta}^-\\ \underline{r}^+\\ \underline{r}^-\end{pmatrix} = \underline{y}\,; \qquad \beta^+, \beta^-, r^+, r^- \geq 0. \qquad (5\text{-}1)$$

After solving the problem, form $\underline{\beta} = \underline{\beta}^+ - \underline{\beta}^-$ and $\underline{r} = \underline{r}^+ - \underline{r}^-$ to recover estimates of the parameter and residual vectors. The optimal value of the objective function is the minimal L_1 norm.

The size of the constraint set in (5-1) is m rows by (2m + 2n) columns. By transforming some of the variables, Barrodale[78] shows that (n - 1) columns of the constraint matrix can be eliminated. To see this, let $\underline{y} = X\underline{\beta} + \underline{r}$

$$\text{or} \quad y_i = \sum_{j=1}^{n} \beta_j \phi_{ij} + r_i, \text{ as before.}$$

Now instead of writing the unrestricted β_j as the difference of two non-negative components as before, define

$$u = \max_j |\beta_j| \geq 0 \Longrightarrow \left[-u \leq \beta_j \leq u\right] \Longrightarrow \left[\beta_j + u \geq 0\right].$$

and let
$$\alpha_j = \beta_j + u \geq 0.$$

Then
$$y_i = \sum_{j=1}^{n} (\alpha_j - u)\phi_{ij} + r_i$$

$$\Longrightarrow y_i = \sum_{j=1}^{n} \alpha_j \phi_{ij} - u \sum_{j=1}^{n} \phi_{ij} + r_i^{+} - r_i^{-}.$$

Finally define
$$\gamma_i = \sum_{j=1}^{n} \phi_{ij},$$

which gives
$$\underline{y} = X\underline{\alpha} - u\underline{\gamma} + I\underline{r}^{+} - I\underline{r}^{-}$$

for the constraint set. The complete problem becomes:

$$\text{minimize} \qquad z = \sum_{i=1}^{n} 0\cdot\alpha_i + 0\cdot u + \sum_{i=1}^{m} 1\cdot r_i^+ + \sum_{i=1}^{m} 1\cdot r_i^-$$

$$\text{subject to} \qquad (\underline{X} \,|\, -\underline{Y} \,|\, I \,|\, -I)\begin{pmatrix} \underline{\alpha} \\ u \\ \underline{r}^+ \\ \underline{r}^- \end{pmatrix} = \underline{Y};\quad \underline{\alpha}, \underline{r}^+, \underline{r}^- \geq \underline{0}, \quad u \geq 0 \qquad (5\text{-}2)$$

The vector $-\underline{Y}$ has replaced the submatrix $-X$ in the constrained matrix for a net savings of $n - 1$ columns.

Now solve (5-2) for $\underline{\alpha}, u, \underline{r}^+, \underline{r}^-$. Then $\underline{r} = \underline{r}^+ - \underline{r}^-$ gives the residuals, while the parameter estimates are given by $\beta_j = \alpha_j - u$. The length of the residual vector (in the L_1 sense) is given by the optimal value of the objective function, as before.

Two comments can be made which apply to either (5-1) or (5-2). The LP problem has no Phase I. Because a <u>unit matrix</u> exists in the constraint matrix, there is an initial basic feasible solution. This implies that there is always an <u>optimal basic feasible solution</u>. Furthermore, the existence of this solution does <u>not</u> depend upon the rank of the matrix X.

Alternate optimal solutions may exist. More will be said about this later.

5.3 Solving the L_1 Problem

The L_1 problem of determining index term weights was set up and solved using (5-1) instead of (5-2). Although (5-2) is more efficient, it was unknown to the author at the time the computer programming was done.

The approximation problem is solved here using three subroutines, one of which is a general purpose SIMPLEX routine. (Barrodale has developed one specialized routine for the L_1 problem). A Fortran IV subroutine for linear programming written by R. J. Clasen[79,80] is used to solve the LP problem. A driver subroutine loads the structural matrix A using the data matrix X, loads the right hand side vector $\underline{b}$, using the known dependent variable vector $\underline{y}$, and finally loads the cost vector $\underline{c}'$, which depends only on the structure of the problem and not on the data.

After the A,$\underline{b}$,$\underline{c}$ data have been loaded by the subroutine, the resulting LP problem is solved using the Clasen subroutine. The solution to the LP problem is related to the solution of the approximation problem by using a follower, or interpretive subroutine, which recovers the unrestricted (as to sign) variables β_j from the optimal non-negative solution variables α_j and u of the LP problem.

Computational experience with the solution of L_1 problems for index term weights has shown that the program is quite fast. For typical problems having 25 rows and 72 columns the average solution time was 3.0 seconds, while for larger problems with 50 rows and 122 columns, the average solution time was 6.0 seconds. This is for the IBM 7094/7044 direct coupled system.

5.4 Example Problems

Figure 5-1A shows the initial full simplex tableau which results when the L_1 problem presented as an example in section 3.34 is set up as an LP problem using formulation (5-1). The submatrix X of Fig. 5-1A is the same as the matrix Z of Fig. 3-2, except that the columns of Z have been permuted to form A. This does not effect the problem solution in any way. This same permuted version of Z also appears as matrix $\bar{X}$ of Fig. 5-2A and Fig. 5-3A. To identify columns of $\bar{X}$ with columns of Z the following table is convenient:

Variables ⟶		β_0	β_1	β_2	β_3	β_4	β_5
Column number	Z	1	2	3	4	5	6
cross references	X	1	4	6	5	2	3

Figure 5-1B shows the optimal tableau for this problem, and Fig. 5-1C gives the solution

$$\hat{u} = \beta_o + \sum_{j=1}^{5} \beta_j T_j = 1T_1 - 1T_2 + 1T_4 - 1T_5$$

which is reconstructed from the optimal LP solution.

The optimal tableau of Fig. 5-1B indicates that an alternate optimal solution is present. Columns indicated with an asterisk are in the optimal basis, while columns paired with the basis columns are marked with 'P'. (Recall that all columns in the structural matrix A

FIGURE 5-3

SAMPLE L_1 PROBLEM - FORMULATION (5-1)

A. Initial Tableau Showing Input Data

$A = (X|-X|I|-I)$ = structural matrix

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	$\underline{b}$
$\underline{c}'$ =	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
A =	1	1	1	0	0	1	-1	-1	-1	0	0	-1	1								-1								-1
	1	1	1	0	0	0	-1	-1	-1	0	0	0		1								-1							-1
	1	0	1	1	1	1	-1	0	-1	-1	-1	-1			1								-1						-1
	1	0	1	0	1	0	-1	0	-1	0	-1	0				1								-1					-1
	1	1	1	0	1	0	-1	-1	-1	0	-1	0					1								-1				1
	1	1	0	1	1	1	-1	-1	0	-1	-1	-1						1								-1			1
	1	1	1	0	1	0	-1	-1	-1	0	-1	0							1								-1		-1
	1	1	1	1	0	0	-1	-1	-1	-1	0	0								1								-1	1

B. Optimal Tableau

Basis	C_j	$\underline{X}_0$	β_0^+	β_4^+	β_5^+	β_1^+	β_3^+	β_2^+	β_0^-	β_4^-	β_5^-	β_1^-	β_3^-	β_2^-	r_1^+	r_2^+	r_3^+	r_4^+	r_5^+	r_6^+	r_7^+	r_8^+	r_1^-	r_2^-	r_3^-	r_4^-	r_5^-	r_6^-	r_7^-	r_8^-
C_j →			0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	1	1/2	1	0	0	0	0	-1/2	-1	0	0	0	0	1/4	0	-3/4	1/4	0	1/2	0	1/4	-1/4	0	3/4	-1/4	0	-1/2	0	-1/4
12	0	1	0	0	0	0	0	-1	0	0	0	0	0	1	-1/2	0	-1/2	1/2	0	0	0	1/2	1/2	0	1/2	-1/2	0	0	0	-1/2
4	0	1	0	0	0	1	0	0	0	0	0	-1	0	0	-1/2	0	1/2	-1/2	0	0	0	1/2	1/2	0	-1/2	1/2	0	0	0	-1/2
9	0	1	-1/2	0	-1	0	0	0	1/2	0	1	0	0	0	-1/4	0	-1/4	-1/4	0	1/2	0	-1/4	1/4	0	1/4	1/4	0	-1/2	0	1/4
17	1	1	-1/2	0	0	0	0	0	1/2	0	0	0	0	0	-1/4	0	3/4	-5/4	1	-1/2	0	-1/4	1/4	0	-3/4	5/4	-1	1/2	0	1/4
5	0	0	1/2	0	0	0	1	0	-1/2	0	0	0	-1	0	-1/4	0	-1/4	3/4	0	1/2	0	-1/4	1/4	0	1/4	-3/4	0	-1/2	0	1/4
27	1	1	1/2	0	0	0	0	0	-1/2	0	0	0	0	0	1/4	0	-3/4	5/4	0	1/2	-1	1/4	-1/4	0	3/4	-5/4	0	-1/2	1	-1/4
22	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1/2	-1	-1/2	1/2	0	0	0	1/2	-1/2	1	1/2	-1/2	0	0	0	-1/2
$\underline{C}_B' B^{-1} \underline{P}_j = Z_j$		3	0	0	0	0	0	0	0	0	0	0	0	0	1/2	-1	-1/2	1/2	1	0	-1	1/2	-1/2	1	1/2	-1/2	-1	0	1	-1/2
$C_j - Z_j$ →			0	0	0	0	0	0	0	0	0	0	0	0	1/2	2	3/2	1/2	0	1	2	1/2	3/2	0	1/2	3/2	2	1	0	3/2
Basis			↑	*	P	*	*	P	↑	P	*	P	P	*		P			*		P			*			P		*	

Alternate Optima (β_0^+, β_0^-)

C. Solution Interpretation

$$\beta_0 = \beta_0^+ - \beta_0^- = 0 - 0 = 0 \qquad r_2 = r_2^+ - r_2^- = 0 - 1 = -1$$

$$\beta_1 = \beta_1^+ - \beta_1^- = 1 - 0 = 1 \qquad r_5 = r_5^+ - r_5^- = 1 - 0 = 1$$

$$\beta_2 = \beta_2^+ - \beta_2^- = 0 - 1 = -1 \qquad r_7 = r_7^+ - r_7^- = 0 - 1 = -1$$

$$\beta_3 = \beta_3^+ - \beta_3^- = 0 - 0 = 0$$

$$\beta_4 = \beta_4^+ - \beta_4^- = 1 - 0 = 1 \qquad L_1(\underline{r}) = Z_0 = 3$$

$$\beta_5 = \beta_5^+ - \beta_5^- = 0 - 1 = -1$$

$$\hat{u} = \beta_0 + \sum_{j=1}^{5} \beta_j T_j = 1T_1 - 1T_2 + 1T_4 - 1T_5$$

Note: Columns in the optimal basis are indicated with an asterisk. Columns out of the basis, but "paired" to columns in the basis are indicated with the letter 'P'.

FIGURE 5-2

SAMPLE L_1 PROBLEM - FORMULATION (5-1) SHOWING ALTERNATE OPTIMAL TABLEAU

A. Alternate Optimal Tableau

Basis	c_j	X_o	1	2	3	4	5	6	7	8	9	10	11	12
	c_j →		0	0	0	0	0	0	0	0	0	0	0	0
Basis	↓	X_o	β_0^+	β_4^+	β_5^+	β_1^+	β_3^+	β_2^+	β_0^-	β_4^-	β_5^-	β_1^-	β_3^-	β_2^-
2	0	2	0	1	-1	0	0	0	0	-1	1	0	0	0
12	0	1	0	0	0	0	0	-1	0	0	0	0	0	1
4	0	1	0	0	0	1	0	0	0	0	0	-1	0	0
7	0	2	-1	0	-2	0	0	0	1	0	2	0	0	0
17	1	0	0	0	1	0	0	0	0	0	-1	0	0	0
5	0	1	0	0	-1	0	1	0	0	0	1	0	-1	0
27	1	2	0	0	-1	0	0	0	0	0	1	0	0	0
22	1	1	0	0	0	0	0	0	0	0	0	0	0	0
$\underline{C}_B' B^{-1} \underline{P}_j = Z_j$		3.	0	0	0	0	0	0	0	0	0	0	0	0
$C_j - Z_j$ →			0	0	0	0	0	0	0	0	0	0	0	0
Basis			P	*	↑	*	*	P	*	P	↑	P	P	*

Basis	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
c_j	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Basis	r_1^+	r_2^+	r_3^+	r_4^+	r_5^+	r_6^+	r_7^+	r_8^+	r_1^-	r_2^-	r_3^-	r_4^-	r_5^-	r_6^-	r_7^-	r_8^-
2	0	0	-1	0	0	1	0	0	0	0	1	0	0	-1	0	0
12	-1/2	0	-1/2	1/2	0	0	0	1/2	1/2	0	1/2	-1/2	0	0	0	-1/2
4	-1/2	0	1/2	-1/2	0	0	0	1/2	1/2	0	-1/2	1/2	0	0	0	-1/2
7	-1/2	0	-1/2	-1/2	0	1	0	-1/2	1/2	0	1/2	1/2	0	-1	0	1/2
17	0	0	1	-1	1	-1	0	0	0	0	-1	1	-1	1	0	0
5	-1/2	0	-1/2	1/2	0	1	0	-1/2	1/2	0	1/2	-1/2	0	-1	0	1/2
27	0	0	-1	1	0	1	-1	0	0	0	1	-1	0	-1	1	0
22	1/2	-1	-1/2	1/2	0	0	0	1/2	-1/2	1	1/2	-1/2	0	0	0	-1/2
$\underline{C}_B' B^{-1} \underline{P}_j = Z_j$	1/2	-1	-1/2	1/2	1	0	-1	1/2	-1/2	1	1/2	-1/2	-1	0	1	-1/2
$C_j - Z_j$	1/2	2	3/2	1/2	0	1	2	1/2	3/2	0	3/2	3/2	2	1	0	3/2
Basis					*					*					*	

Alternate Optima

Note: Columns in the optimal basis are indicated with an asterisk. Columns out of the basis but "paired" to a basis column are indicated with the letter P.

B. Solution Interpretation

$\beta_0 = \beta_0^+ - \beta_0^- = 0 - 2 = -2$

$\beta_1 = \beta_1^+ - \beta_1^- = 1 - 0 = 1$

$\beta_2 = \beta_2^+ - \beta_2^- = 0 - 1 = -1$

$\beta_3 = \beta_3^+ - \beta_3^- = 1 - 0 = 1$

$\beta_4 = \beta_4^+ - \beta_4^- = 2 - 0 = 2$

$\beta_5 = \beta_5^+ - \beta_5^- = 0 - 0 = 0$

$r_2 = r_2^+ - r_2^- = 0 - 1 -1$

$r_7 = r_7^+ - r_7^- = 0 - 2 = -2$

$L_1(\underline{r}) = Z_0 = 3$

$$\hat{u} = \beta_0 + \sum_{j=1}^{5} \beta_j T_j = -2 + T_1 - T_2 + T_3 + 2T_4$$

FIGURE 5-3

SAMPLE L_1 PROBLEM - FORMULATION (5-2)

A. Initial Tableau Showing Input Data

$A = (X|-y|I|-I)$ = structural matrix

$$\underline{C}' = \left[\begin{array}{cccccc|c|cccccccc|cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

$$A = \left[\begin{array}{cccccc|c|cccccccc|cccccccc}
1 & 1 & 1 & 0 & 0 & 1 & -4 & 1 & & & & & & & & -1 & & & & & & & \\
1 & 1 & 1 & 0 & 0 & 0 & -3 & & 1 & & & & & & & & -1 & & & & & & \\
1 & 0 & 1 & 1 & 1 & 1 & -5 & & & 1 & & & & & & & & -1 & & & & & \\
1 & 0 & 1 & 0 & 1 & 0 & -3 & & & & 1 & & & & & & & & -1 & & & & \\
1 & 1 & 1 & 0 & 1 & 0 & -4 & & & & & 1 & & & & & & & & -1 & & & \\
1 & 1 & 0 & 1 & 1 & 1 & -5 & & & & & & 1 & & & & & & & & -1 & & \\
1 & 1 & 1 & 0 & 1 & 0 & -4 & & & & & & & 1 & & & & & & & & -1 & \\
1 & 1 & 1 & 1 & 0 & 0 & -4 & & & & & & & & 1 & & & & & & & & -1
\end{array}\right] \qquad \underline{b} = \left[\begin{array}{c} -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ 1 \end{array}\right]$$

B. Optimal Tableau

Column			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	C_j →		0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Basis	↓	$\underline{X}_0$	α_0	α_4	α_5	α_1	α_3	α_2	u	r_1^+	r_2^+	r_3^+	r_4^+	r_5^+	r_6^+	r_7^+	r_8^+	r_1^-	r_2^-	r_3^-	r_4^-	r_5^-	r_6^-	r_7^-	r_8^-
2	0	4	-1	1	0	0	0	0	0	-1/2	0	3/2	-7/2	3	-1	0	-1/2	1/2	0	-3/2	7/2	-3	1	0	1/2
7	0	2	-1	0	0	0	0	0	1	-1/2	0	3/2	5/2	2	-1	0	-1/2	1/2	0	-3/2	5/2	-2	1	0	1/2
4	0	3	1	0	0	1	0	0	0	-1	0	2	-3	2	-1	0	0	1	0	-2	3	-2	1	0	0
17	1	1	0	0	0	0	0	0	0	1/2	-1	-1/2	1/2	0	0	0	1/2	-1/2	1	1/2	-1/2	0	0	0	-1/2
6	0	1	-1	0	0	0	0	1	0	0	0	2	-3	2	-1	0	-1	0	0	-2	3	-2	1	0	1
22	1	2	0	0	0	0	0	0	0	0	0	0	0	1	0	-1	0	0	0	0	0	-1	0	1	0
5	0	3	-1	0	0	0	1	0	0	-1	0	2	-3	3	-1	0	-1	1	0	-2	3	-3	1	0	1
3	0	2	-1	0	0	0	0	0	0	-1/2	0	5/2	-7/2	3	-2	0	-1/2	1/2	0	-5/2	7/2	-3	2	0	1/2
$\underline{C}_B' B^{-1} \underline{P}_j = Z_j$		3	0	0	0	0	0	0	0	1/2	-1	-1/2	1/2	1	0	-1	1/2	-1/2	1	1/2	-1/2	-1	0	1	-1/2
$C_j - Z_j$ →			0	0	0	0	0	0	0	1/2	2	3/2	1/2	0	1	2	1/2	3/2	0	1/2	3/2	2	1	0	3/2
Basis				*	*	*	*	*	*										*					*	

Alternate Optima

C. Solution Interpretation

$\alpha_0 = 0$ $\qquad \beta_0 = \alpha_0 - u = 0 - 2 = -2$ $\qquad r_5 = r_5^+ - r_5^- = 0 - 1 = -1$

$\alpha_1 = 3$ $\qquad \beta_1 = \alpha_1 - u = 3 - 2 = 1$ $\qquad r_7 = r_7^+ - r_7^- = 0 - 2 = -2$

$\alpha_2 = 1$ $\qquad \beta_2 = \alpha_2 - u = 1 - 2 = -1$

$\alpha_3 = 3$ $\qquad \beta_3 = \alpha_3 - u = 3 - 2 = 1$ $\qquad L_1(\underline{r}) = Z_0 = 3$

$\alpha_4 = 4$ $\qquad \beta_4 = \alpha_4 - u = 4 - 2 = 2$ $\qquad u = 2$

$\alpha_5 = 2$ $\qquad \beta_5 = \alpha_5 - u = 2 - 2 = 0$

$$\hat{u} = \beta_0 + \sum_{j=1}^{5} \beta_j T_j = -2 + T_1 - T_2 + T_3 + 2T_4$$

have a paired column of the opposite sign in formulation (5-1)). Columns not in the optimal basis but having their associated $(c_j - z_j) = 0$, (neglecting columns marked with P) indicate that an alternate optimal solution can be attained with column 7 (β_0^-) in the basis and column 9 (β_5^-) out of the basis. Figure 5-2A shows the tableau for this alternate optimal solution. Note that the solution parameters have changed and the LUPF is different.

Figure 5-3 shows the same problem solved using formulation (5-2). The optimal solution is the same as that given in Figure 5-2 using formulation (5-1).

5.5 The Effects of Alternate Optima

The appearance of alternate optimal solutions to the L_1 approximation problem very simply means that we should be indifferent to the effects of using different estimated LUPF's which might arise from the alternate optima.

Each optimal LUPF gives the same 'best' L_1 fit to the user assigned utilities in the training set, in the sense that $\sum_i |r_i|$ is the same for each LUPF.

A search of the rest of the file with a different LUPF will undoubtedly yield different results, but without using extra information to eliminate the alternate optima, one optimal LUPF is as good as any other. The use of extra information to limit alternate optimal solutions is suggested in chapter 9 as an extension of the present system which might be investigated as a future research problem.

Figure 5-4 gives an example of the different utilities which would be predicted for the various term combinations when two alternate optimal solutions are compared. All 32 combinations of five index terms $T_1,\cdots,T_5$ are listed in Fig. 5-4. (Term T_o is fixed at $T_o = 1$ and hence does not affect the number of combinations.) The utilities which were assigned for the term combinations corresponding to the eight documents in the training set are shown separately. These combinations are numbered 2,3,13,21,25,27,29. Note that two different documents were in the training set with the same index term combination (combination 25). The assigned utilities were different for the two documents (one was relevant, the other was not). Solutions 1 and 2 of Fig. 5-4 show the LUPF's which correspond to the alternate optimal LP solutions illustrated previously in Figs. 5-1 and 5-2. Each of these solutions provide a 'best' (but different) fit to the training set utilities. They also provide different utility predictions for documents outside the training set. In some cases differences in the predicted utilities cause the predicted document relevance category ($u \geq \tau = 0$) to differ. For example, the term combinations 4,8,15,16, 22,28,32 are predicted relevant using solution 1 but non-relevant using solution 2. Combination 17 is predicted non-relevant under solution 1 but relevant under solution 2.

FIGURE 5-4

EFFECTS OF ALTERNATE OPTIMAL SOLUTIONS ON PREDICTED UTILITIES

	T_0	T_1	T_2	T_3	T_4	T_5	User assigned utility	- Predicted Utility - Solution 1	Solution 2
1	1	1	1	1	1	1	-----	0	+1
2	1	1	1	1	1	0	+1	+1	+1
3	1	1	1	1	0	1	-1	-1	-1
4	1	1	1	1	0	0	-----	0	-1
5	1	1	1	0	1	1	-----	0	0
6	1	1	1	0	1	0	-----	+1	0
7	1	1	1	0	0	1	-----	-1	-2
8	1	1	1	0	0	0	-----	0	-2
9	1	1	0	1	1	1	-----	+1	+2
10	1	1	0	1	1	0	-----	+2	+2
11	1	1	0	1	0	1	-----	0	0
12	1	1	0	1	0	0	-----	+1	0
13	1	1	0	0	1	1	+1	+1	+1
14	1	1	0	0	1	0	-----	+2	+1
15	1	1	0	0	0	1	-----	0	-1
16	1	1	0	0	0	0	-----	+1	-1
17	1	0	1	1	1	1	-----	-1	0
18	1	0	1	1	1	0	-----	0	0
19	1	0	1	1	0	1	-----	-2	-2
20	1	0	1	1	0	0	-----	-1	-2
21	1	0	1	0	1	1	-1	-1	-1
22	1	0	1	0	1	0	-----	0	-1
23	1	0	1	0	0	1	-----	-2	-3
24	1	0	1	0	0	0	-----	-1	-3
25	1	0	0	1	1	1	+1,-1	0	+1
26	1	0	0	1	1	0	-----	+1	+1
27	1	0	0	1	0	1	-1	-1	-1
28	1	0	0	1	0	0	-----	0	-1
29	1	0	0	0	1	1	-1	0	0
30	1	0	0	0	1	0	-----	+1	0
31	1	0	0	0	0	1	-----	-1	-2
32	1	0	0	0	0	0	-----	0	-2

Solution 1

$$\hat{u} = T_1 - T_2 + T_4 - T_5$$

Solution 2

$$\hat{u} = -2 + T_1 - T_2 + T_3 + 2T_4$$

5.6 Secondary Feature Extraction

By referring to Figure 5-1A, note that the submatrix X has six columns. Each of these six columns represents a possible term in the LUPF. Five of these columns represent specific <u>index terms</u> which had been previously selected using the information measure of chapter 4.

The optimal tableau shown in Fig. 5-1B indicates that only four (out of a possible six) columns of X (or $-\underline{X}$) are in the optimal basis. Four (out of a possible five) index terms have been assigned to the LUPF shown in Fig. 5-1C. A secondary index term selection has taken place.

This secondary term selection (or feature extraction) process has the effect of <u>discarding automatically</u> index terms (columns) from the basis which are linearly <u>dependent</u> on other terms in the basis.

If the least squares solution were used instead, the linearly dependent columns of X would have to be eliminated <u>before</u> solving the normal equations. The L_1 formulation here eliminates this extra operation.

5.7 More Efficient Algorithms

It can be noted that the parameter vector $\underline{b} = \underline{x}_o$ obtained with the L_1 norm configuration has elements which are integral multiples of 1/2, i.e., $b_i = \pm n/2$. This effect is obviously dependent on properties of the inverses of matrices whose elements are all +1, -1 or zero, and of the integral properties of the right hand side vector (the utilities).

The properties of $\underline{x}_0$ suggest that perhaps the LP problem for this type of matrix can be solved with a transportation or network type of algorithm. Investigation of this was outside the scope of this work.

6.0 DETERMINATION OF THE OPTIMAL BRS

6.1 Scope and Organization

The optimal BRS is a set of searching instructions which <u>retrieves</u> from a file only those documents having a <u>predicted</u> utility greater than or equal to a given utility threshold.

The optimal BRS is derived from the LPBI which is formed by thresholding the document LUPF.

This chapter discusses mathematical properties of the LPBI and of its solutions. A composite algorithm is presented which finds all the solutions to the LPBI and groups these into solution families which are mutually disjoint. This composite algorithm is based on visiting the nodes of a binary tree in search of possible solutions to the inequality. It is called the Tree Pruning Algorithm (TPA), and uses a branch-and-exclude technique which allows all solutions to be found without constructing or exploring the entire binary solution tree.

The composite TPA can be broken down into two parts. The first part is a node-visiting sub-algorithm. Here decisions are made (after visiting a tree node) about which nodes of the tree to exclude from future visits. The second part of the TPA is a visit-scheduling sub-algorithm which controls the sequencing of node visits. This sub-algorithm guarantees that each non-excluded node is visited once and only once in a defined order. It also keeps node records necessary for use by the node-visiting sub-algorithm. The visit-scheduling sub-algorithm is necessary to implement the TPA on a digital computer.

The concepts and theory pertinent to solving a LPBI by a node-visiting method have been given elsewhere by Hammer and Rudeanu[81,82,83]. Most of the mathematical details presented here are also from these references. An exception is section 6.323. Here some proofs are presented which are related to transformations used to solve the LPBI. These proofs are not given by Hammer and Rudeanu. Background theoretical results and details of the node-visiting sub-algorithm are presented in the first part of this chapter, up to and including section 6.5.

The visit scheduling sub-algorithm is the Author's contribution to the TPA. It is a modified form of a pre-order traversal algorithm for binary trees. This sub-algorithm allows dynamic visit-scheduling as portions of the binary tree are sequentially excluded from further consideration. Development of this sub-algorithm begins in section 6.6.

The operation of the composite TPA is illustrated with examples, and computational experience with a Fortran IV program is discussed.

The use of the LPBI solution families to retrieve documents is discussed near the end of the chapter.

6.2 The LPBI Arising from the Document LUPF

It is assumed that a LUPF exists which adequately expresses the utility of documents in the file as a linear combination of selected index term weights, i.e.

$$\hat{u} = \sum_{j=0}^{n} a_j T_j, \quad \left[\begin{array}{l} T_j \varepsilon \{0,1\} \\ -\infty < a_j < \infty \end{array} \right.$$

which becomes a pseudo-Boolean inequality when thresholded;

$$\sum_{j=1}^{n} a_j T_j \geq (\tau - a_o)$$

$$-\infty < \tau < \infty .$$

After conversion of the coefficients a_j and the right hand side $(\tau - a_o)$ to integers γ_j and δ by a scaling and truncating process,

$$\sum_{j=1}^{n} \gamma_j T_j \geq \delta, \quad \left[\begin{array}{l} T_j \varepsilon \{0,1\} \\ \delta, \gamma_j \varepsilon \{I\} \end{array} \right. \tag{6-1}$$

where I is the set of all integers.

For all further results in this chapter the LPBI will be assumed to have integer coefficients. This represents no loss of generality/because by scaling all coefficients and right hand side, and then dropping the fractional parts, if any, the coefficients can be converted to integers with any desired degree of accuracy.

All solutions of inequality (6-1) are 0/1 vectors $\underline{T}_k = (T_{k1}, T_{k2}, \cdots T_{kn})$. There are at most 2^n vectors $\underline{T}_k$ satisfying (6-1). Solution by enumeration is always possible but becomes impractical for all but small problems. Moreover, solution by enumeration does not group solution vectors into families.

Grouping of solution vectors into families is important for two reasons:

(a) one solution family provides a compact mathematical representation of many solution vectors;

(b) the solution families are meaningful in the modeling of document retrieval systems. More will be said about this in section 6.74.

6.3 Properties of the LPBI and Its Solutions

As a prelude to developing an algorithm to solve the inequality (6-1) for all of its solution vectors and/or families of solution vectors, it is necessary to investigate a more general form of (6-1).

6.31 General Form of the LPBI

Let the linear pseudo-Boolean inequality in its general form be defined by:

$$\sum_{j=1}^{n} z_j^{\alpha_j} \gamma_j \geq \delta \qquad (6\text{-}2)$$

where α_j, γ_j and δ are given parameters with

$$\alpha_j \varepsilon \{0,1\} \qquad j=1,2,\cdots,n.$$

$$\gamma_j, \delta \in \{I\} \qquad \text{the set of all integers}$$

and where $\underline{z} = (z_j^1)$, $j=1,2,\cdots,n$

is a solution vector, with $z_j^1 \varepsilon \{0,1\}$.

The exponents are used to indicate Boolean complements with the following conventions:

$$z_j^o \equiv \overline{z}_j, \text{ the complement of } z_j^1; \tag{6-3}$$

$$z_j^1 \equiv z_j;$$

$$z_j^o = 1 - z_j^1;$$

$$(z_j^1)^{\alpha_j} = z_j^{\alpha_j} .$$

As a consequence of this exponent notation, note that:

$$\left(z_j^{a_j}\right)^{\alpha_j} = z_j^1 \text{ if } a_j = \alpha_j$$

$$= z_j^o \text{ if } a_j \neq \alpha_j .$$

The inequality (6-1) arising from the LUPF is equivalent to (6-2) if all $\alpha_j = 1$. The algorithm developed in this chapter will solve form (6-2) of the LPBI.

The adjective pseudo-Boolean implies that while the variables z_j of (6-2) are binary valued, the coefficients are not, and hence the function

$$L(z_1,z_2,\cdots,z_n) = L(\underline{z}) = \sum_{j=1}^{n} z_j^{\alpha_j}\gamma_j$$

is a mapping of the binary vector $\underline{z}$ into the set of positive or negative integers. This is in distinction to a Boolean function $f(\underline{z})$ which would map the binary vector $\underline{z}$ into the binary set $\{0,1\}$.

6.32 Canonical Form of the LPBI

Before solving the inequality (6-2) it is necessary to reduce it to a standard or canonical mathematical form.

The canonical form is defined by

$$\sum_{j=1}^{n} c_j x_j^1 \geq d; \; (c_j, d) \in \{I\} \qquad (6\text{-}4)$$

where $\underline{x} = (x_j^1)$, $j=1,\cdots,n$ is the solution vector and $c_1 \geq c_2 \geq \cdots \geq c_n > 0$. This form has all positive coefficients c_j, ranked by order of magnitude. In addition, no complemented variables x_j appear.

6.321 Transformation of Parameters of the LPBI from the General Form to the Canonical Form. The transformation from (6-2) to (6-4) proceeds in two stages.

First, all negative coefficients are eliminated by the following transformation, (and all γ_j are relabeled e_j):

$$\gamma_j > 0 \Longrightarrow (\gamma_j \leftarrow e_j; \; \alpha_j \leftarrow a_j)$$

$$\gamma_j < 0 \Longrightarrow (\gamma_j \leftarrow -e_j; \; \alpha_j \leftarrow a_j = 1 - \alpha_j) \qquad (6\text{-}5)$$

$$d \leftarrow \delta - \sum_{(\gamma_j < 0)} \gamma_j$$

where $a \leftarrow b$ is read "a is replaced by b". At this point a new inequality may be defined by:

$$\sum_{j=1}^{n} y_j^1 e_j \geq d \qquad (6\text{-}6)$$

$$y_j^1 \in \{0,1\}$$

$$(e_j, d) \in \{I\}$$

$$e_j > 0 .$$

The coefficients e_j are next permuted and relabeled so they are in descending order as specified by (6-4). We define a transformation from e_j to c_j by

$$k \leftarrow P(j) \qquad (6\text{-}7)$$

$$c_j' \leftarrow e_k$$

$$j=1,2,\cdots,n.$$

where $P(j)$ is a permutation which puts coefficients e_j in descending order. This completes the transformation of parameters to (6-4) from (6-2).

For example, consider a pseudo-Boolean inequality whose parameters consist of:

j	γ_j	α_j
1	-2	1
2	-3	0
3	5	0
4	-1	1
5	2	1

$\delta = 0$

Eliminating negative coefficients results in new parameters.

j	e_j	a_j	
1	2	0	
2	3	1	
3	5	0	
4	1	0	$d = 6$
5	2	1	

Permuting and relabeling coefficients e_j as c_j gives:

j	c_j	$P(j)$	
1	5	3	
2	3	2	
3	2	1	$d = 6$.
4	2	5	
5	1	4	

The permutation $P(j)$ is obtained from a sort of the e_j. If the indices j are sorted along with the e_j; the result is $P(j)$. Note that the α_j are transformed into the a_j when the negative coefficients are eliminated. Permuting and relabeling does not modify the a_j.

We will be concerned with solutions $\underline{x}_k = (x^1_{kj})$ of the canonical form (6-4). The approach is to find solutions to this form, then perform appropriate inverse transformations on these solutions to get vectors $\underline{z}_k = (z^1_{kj})$ which satisfy inequality (6-2).

6.322 Transformation of Solutions of the LPBI from the Canonical Form to the General Form. We have defined three inequalities by performing the preceding transformations on the parameters. These are repeated below for comparison.

$$\sum z_j^{\alpha_j} \gamma_j \geq \delta \tag{6-2}$$

$$\sum y_j^1 e_j \geq d \tag{6-6}$$

$$\sum x_j^1 c_j \geq d \tag{6-4}$$

Solutions to (6-4) will be appropriately transformed so they become solutions to (6-6) and finally (6-2). These inverse transformations proceed in two steps as follows:

(a) from $\underline{x}$ to $\underline{y}$ where

$$\begin{array}{c} k \leftarrow P(j) \\ y_k^1 \leftarrow x_j^1 \\ j=1,2,\cdots,n; \end{array} \tag{6-8}$$

(b) from $\underline{y}$ to $\underline{z}$ where

$$\begin{array}{l} a_j = 1 \Rightarrow z_j^1 \leftarrow 1 - y_j^1 \\ a_j = 0 \Rightarrow z_j^1 \leftarrow y_j^1 \ ; \end{array} \tag{6-9}$$

that is: $z_j^1 \leftarrow y_j^{a_j}$.

The transformations defined above can be depicted as shown below.

$$\begin{matrix} z_j, \alpha_j \\ \delta \end{matrix} \xrightarrow{(6\text{-}5)\ (6\text{-}7)} \begin{matrix} c_j, a_j \\ d \end{matrix}$$

$$\left\{ \underline{z}_k \,\middle|\, \sum_{j=1}^{n} z_{kj}^{\alpha_j}\gamma_j \geq \delta \right\} \xleftarrow{(6\text{-}8),\ (6\text{-}9)} \left\{ \underline{x}_k \,\middle|\, \sum_{j=1}^{n} x_{kj}c_j \geq d \right\}$$

The solution transformation has as its object set all solutions of the canonical form (6-4), and as its image set all solutions of the general form (6-2).

6.323 <u>Some Proofs of Results Related to the Transformations</u>. It is easy to prove that a binary vector $\underline{z}_k = (z_{kj})$ is a solution to inequality (6-2) if and only if the corresponding vector $\underline{x}_k = (x_{kj})$ is a solution to inequality (6-4) when (6-5) and (6-7) are used to transform the coefficients, and (6-8) and (6-9) are used to transform $\underline{x}_k$ to $\underline{z}_k$. That is,

$$\left(\sum_{j=1}^{n} z_j^{\alpha_j}\gamma_j \geq \delta \right) \Longleftrightarrow \left(\sum_{j=1}^{n} x_j^{1}c_j \geq d \right). \qquad (6\text{-}10)$$

To show this it is convenient to establish two preliminary results. First, note that we need consider only transformations from (6-6) to (6-2) instead of from (6-4) to (6-2). This is because a solution $\underline{x}_k$ to (6-4) is always transformed by (6-8) into a solution $\underline{y}_k$

of (6-6). Recall that transformation (6-8) is merely a permutation of coefficients, i.e.

$$x_j c_j = y_{P(j)} e_{P(j)}, \quad j=1,2,\cdots,n$$

$$\Longrightarrow \quad \sum_{j=1}^{n} x_j^1 c_j = \sum_{j=1}^{n} y_j^1 e_j .$$

$$\Longrightarrow \left(\sum_{j=1}^{n} x_j^1 c_j \geq d\right) \Longleftrightarrow \left(\sum_{j=1}^{n} y_j^1 e_j \geq d\right). \tag{6-11}$$

Another preliminary result is derived from the assumption with no loss of generality that the first p coefficients γ_j are <u>positive</u> and the last $(n - p)$ coefficients γ_j are <u>negative</u>, i.e.

$$\begin{aligned} \gamma_j &> 0; \; j=1,2,\cdots,p \\ \gamma_j &< 0; \; j=p+1,\cdots,n. \end{aligned} \tag{6-12}$$

Then after the transformation (6-5), note that we can conveniently express e_j, a_j and d in terms of γ_j, α_j and δ as follows:

$$\begin{aligned} & d = \delta - \sum_{j=p+1}^{n} \gamma_j \\ & \left.\begin{aligned} e_j &= \gamma_j \\ a_j &= \alpha_j \end{aligned}\right\} j=1,2,\cdots,p \\ & \left.\begin{aligned} e_j &= -\gamma_j \\ a_j &\neq \alpha_j \end{aligned}\right\} j=p+1,\cdots,n . \end{aligned} \tag{6-13}$$

Now by using (6-3), it follows that:

$$\left.\begin{aligned} \left(y_j^{a_j}\right)^{\alpha_j} &= y_j^1 \end{aligned}\right\} \quad j=1,2,\cdots,p \tag{6-14}$$

$$\left.\begin{aligned} \left(y_j^{a_j}\right)^{\alpha_j} &= y_j^o = 1 - y_j^1 \\ \Longrightarrow \quad y_j^1 &= 1 - \left(y_j^{a_j}\right)^{\alpha_j} \end{aligned}\right\} \quad j=p+1,\cdots,n .$$

By using the above results, the first half of (6-10), i.e.

$$\left(\sum_{j=1}^{n} y_j^1 e_j \geq d\right) \Longrightarrow \left(\sum_{j=1}^{n} z_j^{\alpha_j} \gamma_j \geq \delta\right) \tag{6-15}$$

is proven as follows:

$$\sum_{j=1}^{n} y_j^1 e_j \geq d \Longrightarrow \sum_{j=1}^{p} y_j^1 e_j + \sum_{p+1}^{n} y_j^1 e_j + \sum_{p+1}^{n} \gamma_j \geq \delta$$

$$\Longrightarrow \sum_{j=1}^{p} \left(y_j^{a_j}\right)^{\alpha_j} \gamma_j + \sum_{p+1}^{n} \left[1 - \left(y_j^{a_j}\right)^{\alpha_j}\right](-\gamma_j) + \sum_{p+1}^{n} \gamma_j \geq \delta$$

$$\Longrightarrow \sum_{j=1}^{p} \left(y_j^{a_j}\right)^{\alpha_j} \gamma_j \geq \delta$$

and using the fact that $z_j^1 = y_j^{a_j}$ from (6-9) we have the desired result.

Next we would like to prove the second part of (6-10) which is the converse of (6-15), i.e.

$$\left(\sum_{j=1}^{n} z_j^{\alpha_j}\gamma_j \geq \delta\right) \Longrightarrow \left(\sum_{j=1}^{n} y_j^1 e_j \geq d\right) . \tag{6-16}$$

However, this is equivalent to showing that

$$\left(\sum_{j=1}^{n} y_j^1 e_j < d\right) \Longrightarrow \left(\sum_{j=1}^{n} z_j^{\alpha_j}\gamma_j < \delta\right),$$

and this result can be shown by exactly the same technique used to prove (6-15).

Note also that the transformation (6-9) from $\underline{y}$ to $\underline{z}$ is one-to-one, i.e.

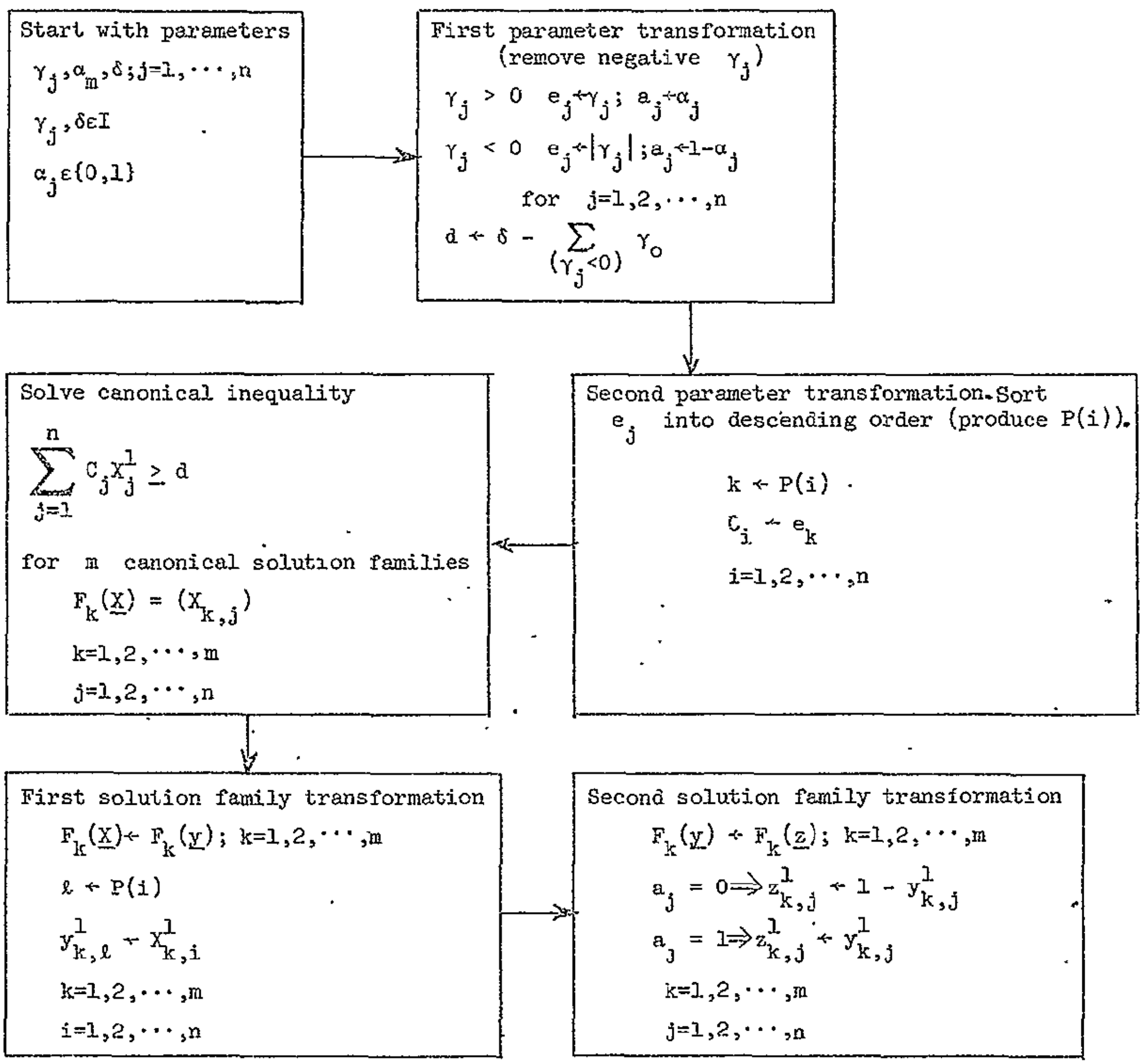
$$(\underline{y}_1 \neq \underline{y}_2) \Longleftrightarrow (\underline{z}_1 \neq \underline{z}_2). \tag{6-17}$$

This is obvious since (6-9) simply complements certain fixed elements of $\underline{y}$ to get $\underline{z}$.

Results (6-10) and (6-17) are important because they guarantee that all solutions to the original inequality (6-2) will be found by first transforming the parameters using (6-5) and (6-7) to get the canonical inequality (6-4); solving this inequality for all its solutions and transforming these solutions back. These transformations are summarized in Fig. 6-1.

FIGURE 6-1

FLOW CHART SHOWING TRANSFORMATIONS INVOLVED IN THE SOLUTION OF A LINEAR PSEUDO-BOOLEAN INEQUALITY



As an example of inverse transformations of solutions, observe that $\underline{x} = (0,1,1,0,1)$ is a solution to the canonical inequality used previously in section 6.321 as an example: $\underline{x}$ transforms to $\underline{y} = (1,1,0,1,0)$ which transforms to $\underline{z} = (0,1,1,0,0)$. This last vector satisfies the original inequality, since $-2(0^1) - 3(1^0) + 5(1^0) - 1(0^1) + 2(0^1) = 0$. (Recall from (6-3) that $z^0 = \bar{z}$.)

Another result which will be useful later to relate values of $\sum_j y_j e_j$ to $\sum_j \gamma_j z_j^{\alpha_j}$ before and after transformation (6-9) is given below

$$\sum_{j=1}^{n} x_j^1 c_j - \sum_{j=1}^{n} z_j^{\alpha_j} \gamma_j = g = - \sum_{(\gamma_j<0)} \gamma_j \geq 0 . \qquad (6\text{-}18)$$

This is easily proven by using preliminary results (6-11) through (6-14) and (6-9) which gives:

$$\sum_{1}^{n} z_j^{\alpha_j} \gamma_j = \sum_{1}^{n} \left(y_j^{a_j}\right)^{\alpha_j} \gamma_j = \sum_{1}^{p} y_j^1 e_j + \sum_{p+1}^{n} y_j^0 (-e_j)$$

$$= \sum_{1}^{p} y_j^1 e_j + \sum_{p+1}^{n} (1 - y_j^1)(-e_j)$$

$$= \sum_{1}^{n} y_j^1 e_j + \sum_{p+1}^{n} \gamma_j$$

$$= \sum_{1}^{n} y_j^1 e_j - g .$$

As a corollary to this we can state that the inverse transformation (6-9) is order-preserving, i.e.

$$\left(\sum_{j=1}^{n} y^{1}_{mj} e_j > \sum_{j=1}^{n} y^{1}_{\ell j} e_j\right) \quad \text{for} \quad m \neq \ell$$
$$\Longrightarrow \left(\sum_{j=1}^{m} (z_{mj})^{\alpha_j} \gamma_j > \sum_{j=1}^{n} (z_{\ell j})^{\alpha_j} \gamma_j\right). \tag{6-19}$$

6.33 Families of Solutions.

A set $\sum$ of solution vectors formed from a given solution vector $\underline{z}_0 \doteq (z^1_{01}, z^1_{02}, \cdots, z^1_{0n})$ and a set of indices $I \subseteq \{1,2,\cdots,n\}$ is called a <u>family of solutions</u>. All members of the set match the solution vector $\underline{z}_0$ at the indices in I and are free to vary at all indices not in I.

For example, $\underline{z}_0 = (0,0,0,1,0)$ is one solution of the example. Let $I = \{1,2,3\}$. The set $\sum(\underline{z}_0, I)$ of solutions contains four solution vectors (including $\underline{z}_0$)

$$(0,0,0,0,0)$$
$$(0,0,0,0,1)$$
$$(0,0,0,1,0)$$
$$(0,0,0,1,1)$$

This family can also be noted as $F = (0,0,0,-,-)$ where $(-)$ indicates either 0 or 1.

If $\sum$ contains only one vector, namely $\underline{z}_0$, it is said to be a <u>degenerate family of solutions</u>. The number of vectors in $\sum$ is given by 2^{n-r} where r is the number of fixed variables (elements) in I.

A group of solution families $\sum_1, \sum_2, \ldots, \sum_n$ is <u>disjoint</u> if each solution vector belongs to one and only one solution family.

Our goal is to find all solution points $\underline{z}_i$ to the inequality (6-2) grouped together into families. It can be shown (see section 6.352) that the method used to group solution points into families results in <u>mutually disjoint</u> solution families.

Families of solutions will be found to the canonical inequality (6-4), and these <u>families</u> will be transformed to solutions of (6-2), using the inverse transformations (6-8) and (6-9). A <u>family of solutions</u> is transformable by (6-8) and (6-9) with the obvious convention that in (6-9) if $y_j = (-)$, then $z_j \leftarrow y_j = (-)$ irrespective of whether $a_j = 0$ or $a_j = 1$.

6.34 The Relationship between Binary Trees and Solutions of a LPBI

Certain isomorphisms exist between binary trees[84] and solutions to pseudo-Boolean inequalities. These relationships prove invaluable for developing algorithms to solve inequalities and to visualize the solution process.

6.341 <u>Isomorphism of Tree Paths to Possible Solutions</u>. Each possible solution to a pseudo-Boolean inequality may be pictured as a <u>path</u>

through a binary solution tree. This is illustrated in Fig. 6-2A for the inequality

$$3x_1 + 2x_2 + x_3 \geq 4 .$$

Starting from the root node r, if we proceed to the left to node a, then $x_1 = 0$. If we go to b from r, then $x_1 = 1$. This takes us to stage 1. To go to stage 2, we can move to c or d from node a, or from node b to either node e or node f. The stage of a node in the solution tree is the number of levels which the node is removed from the root node. There are $n+1$ stages in the complete solution tree associated with an inequality having n variables.

If we traverse the tree from the root node r to node i in the path $r \rightarrow a \rightarrow d \rightarrow i$, we have enumerated one of the $2^3 = 8$ binary vectors $\underline{x} = (x_1,x_2,x_3) = (0,1,0)$. A move along a left branch from one stage to the next implies that the variable x_i associated with that stage is to be set at zero. A move to the right implies that the variable is to be set at 1.

By traversing a path from the root to each of the terminal nodes (leaves) of the tree, each binary vector $\underline{x}$ can be enumerated. Each $\underline{x}$ could be tested to find only the $\underline{x}^*$ which are solutions to the inequality. We conclude that each path from the root node to a terminal (leaf) node is isomorphic to a possible solution point $\underline{x}^*$.

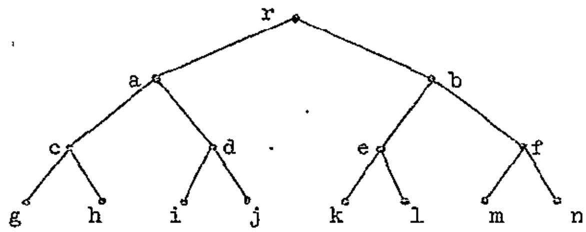
By inspection, nodes l,m and n represent solutions to the inequality.

FIGURE 6-2

SOLUTION TREE AND ASSOCIATED DATA FOR A SIMPLE INEQUALITY

$$3X_1 + 2X_2 + X_3 \geq 4$$

A. Solution tree



Stage	Fixed variable
0	
1	X_1
2	X_2
3	X_3

Value	$\Sigma C_j X_j$	0	1	2	3	3	4	5	6
Binary	X_1	0	0	0	0	1	1	1	1
Vectors	X_2	0	0	1	1	0	0	1	1
	X_3	0	1	0	1	0	1	0	1

B. Partial Path Records and Partial Inequalities Associated with Tree Nodes

Node	Stage	Partial Path record	Partial Inequality
r	0	(-,-,-)	$3X_1 + 2X_2 + X_3 \geq 4$
a	1	(0,-,-)	$2X_2 + X_3 \geq 4$
b	1	(1,-,-)	$2X_2 + X_3 \geq 1$
c	2	(0,0,-)	$X_3 \geq 4$
d	2	(0,1,-)	$X_3 \geq 2$
e	2	(1,0,-)	$X_3 \geq 1$
f	2	(1,1,-)	$X_3 \geq -1$
g	3	(0,0,0)	----------------
h	3	(0,0,1)	----------------
i	3	(0,1,0)	----------------
j	3	(0,1,1)	----------------
k	3	(1,0,0)	----------------
l	3	(1,0,1)	----------------
m	3	(1,1,0)	----------------
n	3	(1,1,1)	----------------

6.342 Isomorphism of Tree Nodes to Partial Path Records and Partial Inequalities. Associated with each node in the tree is a set of fixed binary variables and a set of arbitrary binary variables.

The fixed set of variables represents a partial path record (PPR) from the root node to any other node in the tree. PPR's become complete path records when the path is traced from the root to the terminal (leaf) nodes. See Fig. 6-2B for an illustration. The set of arbitrary variables are those necessary to specify a complete path record from a PPR. For example, at node d, the fixed variables are x_1 and x_2, while x_3 is arbitrary.

A partial inequality (PIN) can also be associated with each node in the solution tree. The variables in these PIN's are those in the set of arbitrary variables, while the set of fixed variables and their coefficients are absorbed into the right hand side of the PIN.

At any p^{th} stage node there are p fixed variables and $(n-p)$ arbitrary variables. The PIN associated with a p^{th} stage node is given by:

$$\sum_{j=p+1}^{n} c_j x_j \geq \left[d - \sum_{j=1}^{p} c_j x_j \right].$$

As an example, node e of Fig. 6-2A has an associated PIN given by:

$$1x_3 \geq 4 - [3(1) + 2(0)] \implies x_3 \geq 1.$$

Fig. 6-2B lists the partial PPR's and PIN's associated with all nodes of the solution tree shown in Fig. 6-2A.

It is possible to construct a binary solution tree for any pseudo-Boolean inequality, whether it is in general or canonical form. Fig. 6-3 shows a solution tree for an inequality in a general form. Fig. 6-4 shows the solution tree for the same inequality after transformation to canonical form.

The canonical form solution tree has special properties which enable families of solutions to be built up automatically from special types of solution tree paths known as <u>basic solution paths</u> (BSP's). These will be discussed extensively in the following sections.

6.35 Solutions of the Canonical Form

Fig. 6-4 shows the solution tree associated with the canonical form of the inequality used as an example in section 6.321. For the canonical inequality all solution values are bounded between 0 and $\sum_1^n c_j$. There are no negative values. There are 19 solutions to the canonical inequality, just as there were to the original inequality.

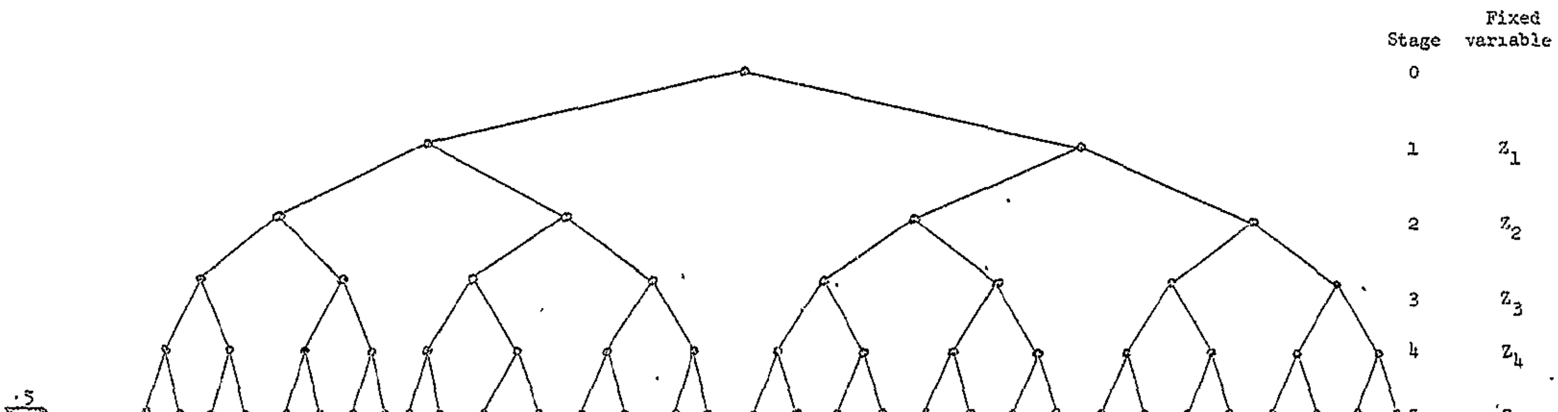
6.351 <u>Basic Solutions.</u> Of the 19 solution vectors $\underline{x}$, seven have special properties. These solutions are called <u>basic solutions</u>. They are formally defined as follows.

A <u>basic solution</u> to the canonical inequality (6-4) is a solution $\underline{x}^* = (x_1^*, x_2^*, \cdots, x_n^*)$ such that for each index i with $x_i = 1$ the vector $(x_1^*, \cdots, x_{i-1}^*, 0, x_{i+1}^*, \cdots, x_n^*)$ is <u>not</u> a solution of (6-4).

FIGURE 6-3

BINARY SOLUTION TREE ASSOCIATED WITH AN LPBI IN GENERAL FORM

$$-2Z_1 - 3\bar{Z}_2 + 5\bar{Z}_3 - Z_4 + 2Z_5 \geq 0$$



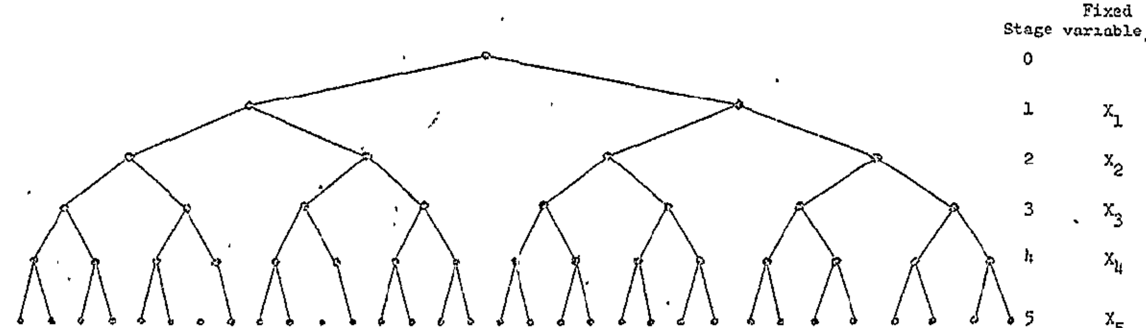
$\sum_{j=1}^{5} Z_j^{\alpha j}\gamma_j \rightarrow$	2	4	1	3	-3	-1	-4	-2	5	7	4	6	0	2	-1	1	0	2	-1	1	-5	-3	-6	-4	3	5	2	4	-2	0	-3	-1
Path vectors Z_1^1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Z_2^1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Z_3^1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
Z_4^1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
Z_5^1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Path number $\rightarrow$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

NOTE: Values of $\sum_{j=1}^{5} Z_j^{\alpha j}\gamma_j$ which satisfy the inequality are circled.

FIGURE 6-4

BINARY SOLUTION TREE ASSOCIATED WITH AN LPBI IN CANONICAL FORM

$$5X_1 + 3X_2 + 2X_3 + 2X_4 + X_5 \geq 6$$



$\sum_{j=1}^{5} X_j^i c_j$	0	1	2	3	2	3	4	5	3	4	5	6	5	6	7	8	5	6	7	8	7	8	9	10	8	9	10	11	10	11	12	13
Basic Solutions												*		*	*			*	*		*				*							
Path Vectors X_1^i	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
X_2^i	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
X_3^i	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
X_4^i	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
X_5^i	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Path Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Notes; Values of $\sum_{j=1}^{5} X_j^i c_j$ which satisfy the inequality are circled.

Basic solutions are marked with *.
Solution families are shown in brackets

6.352 Canonical Solution Families. Given a basic solution $\underline{x}_k^*$ it is possible to define a family of solutions $F_k = \sum(\underline{x}_k^*, I_k)$ in a special manner which exploits the minimal property of the basic solution.

A solution family $F_k = \sum(\underline{x}_k^*, I_k)$ constructed from a basic solution $\underline{x}_k^*$ using the following rules will be called a canonical solution family. Let $\ell, (1 \leq \ell \leq n)$ be the last index for which $x_{ki}^* = 1$, where $\underline{x}_k^* = (x_{k1}^*, x_{k2}^*, \cdots, x_{kn}^*)$ is a basic solution. I_k is then defined to be the set of all indices $i \leq \ell$.

The basic solution is a minimal solution vector $\underline{x}^*$, in the sense that changing any of the variables from 1 to 0 gives a new vector $\underline{x}$ which is not a solution. It is defined only for the canonical form of the LPBI, where all coefficients are positive and all variables are uncomplemented.

In terms of the solution tree, a basic solution corresponds to a solution path through the tree which does not remain a solution path if any right branch is changed to a left branch. In Fig. 6-4, the basic solutions corresponds to tree paths numbered 12,14,15,18,19,21 and 25. A path through the tree corresponding to a basic solution will be referred to as a basic solution path (BSP).

Referring to Fig. 6-4, path number 21 through the binary tree corresponds to solution vector $x_{21}^* = (1,0,1,0,0)$. This solution is basic and path number 21 is a BSP. It can be made into a canonical solution family by allowing arbitrary values for the last two 0-valued vector elements. We can denote this family by $F_{21} = \sum(x^*_{21}, I_{21}) = (1,0,1,-,-)$, where $I_{21} = \{1,2,3\}$.

Canonical solution family F_{21} contains $2^{n-\ell} = 2^{5-3} = 4$ solution vectors as members. These are shown as paths numbered 21-24. The BSP is seen to be the left-most tree path in the family. Some canonical solution families have only one member (the BSP) and are said to be <u>degenerate solution families</u>. In Fig. 6-4, paths numbered 12 and 14 are families of this type.

It can be seen that by knowing only the basic solutions that all other solutions to the canonical inequality can be enumerated. This is formalized by the following result which has been proven by Hammer and Rudeanu(85).

<u>Every solution to the canonical inequality belongs to one and only one canonical solution family</u>.

Because the inverse transformation of canonical solutions is one-to-one (see (6-17)), the above result holds after the transformation. Thus, when the <u>canonical</u> solution families are subjected to the inverse transformations (6-8) and (6-9), we get mutually disjoint solution families to the <u>original inequality</u>.

The problem of solving the pseudo-Boolean inequality is now reduced to the problem of identifying all basic solutions of the canonical inequality. This will be the subject of the next section.

6.36 Summary of Solution Procedure for the LPBI

Section 6.35 shows that the solutions to the LPBI (6-2) may be obtained in mutually disjoint families by the following procedure:

(a) transform the original inequality to canonical form;

(b) determine all basic solutions to the canonical form;

(c) construct canonical solution families using each basic solution;

(d) inversely transform the canonical solution families and get solution families to the original LPBI (6-2).

6.4 Determining Basic Solutions of the LPBI by Searching the Binary Solution Tree

6.41 Preview of the Tree Pruning Algorithm (TPA).

The method used to determine basic solutions of the canonical inequality is based on finding all BSP's in the associated binary solution tree. This method relies upon systematically 'visiting' nodes of the tree, starting at the root node and moving in a downward direction toward the terminal (leaf) nodes. When a node is 'visited', the parameters of the associated PIN are examined. This gives information about which nodes to visit next.

For each node visited, it may be possible to eliminate further downward motion in the tree through one of the following two devices:

(a) by determining that no BSP can exist using a branch directed down to the left, right (or both) of the current node;

(b) by enumerating all complete BSP's which employ branches directed down to the left, right (or both) of the current node. This makes further downward movement unnecessary.

When all downward paths through the solution tree have been blocked by (a) or (b), it follows that all BSP's have been found, and the node visiting operation stops.

The elimination of downward (away from the root) movements in the tree through results obtained higher up (closer to the root) in the tree can be called a 'branch-and-exclude' scheme. The subtree whose nodes are actually visited is then a small segment of the original solution tree. This subtree can be considered to arise from the original tree by a branch-cutting or <u>pruning</u> operation. For this reason the final algorithm developed is called a <u>tree pruning algorithm</u> (TPA).

At a given node, the decision to prune and/or to enumerate BSP's is based on a classification scheme to be applied to the parameters of the PIN associated with the node. The classification scheme is due to Hammer and Rudeanu and is discussed in section 6.52.

When they are identified, complete BSP's are constructed using both the PIN and the PPR at any given node. This is discussed in section 6.51.

Development of the TPA can be broken down logically into two parts. Definition of what is done when a node is visited is one part. The other part is concerned with the scheduling of node visits. Although these two logical parts are linked (node visits can alter the schedule of remaining visits), it will be convenient to consider the node visiting portion first.

Section 6.5 provides theory and methods relating to what is done at an individual node when it is visited. This includes construction of BSP's and pruning of the solution tree.

The scheduling and record keeping details related to node visits are deferred to section 6.6.

6.5 Solution Construction and Node Visits

6.51 Constructing Complete BSP's from Partial BSP's

As nodes in the subtree are visited, the PPR is maintained. Thus suppose at some node currently being visited, a basic solution to the PIN is identified by the scheme to be presented in section 6.52. Then the complete BSP consists of two parts and is constructed in the following manner.

The first part of the complete BSP is the PPR to the current node. The second part is the basic solution of the PIN associated with the current node.

These remarks may be formalized by the following results (see Hammer and Rudeanu[86]).

(A) Let $(x_1^*, x_2^*, \cdots, x_p^*, x_{p+1}^*, \cdots, x_n^*)$ be a basic solution of the canonical inequality (6-4). Then $(x_{p+1}^*, \cdots, x_n^*)$ is a basic solution of the inequality

$$\sum_{j=p+1}^{n} C_j x_j \geq d - \sum_{k=1}^{p} C_k x_k^*.$$

(B) If $(x^*_{p+1}, x^*_{p+2}, \cdots, x^*_n)$ is a basic solution of the inequality

$$\sum_{j=p+1}^{n} C_j x_j \geq d,$$

then $(0, \cdots, 0, x^*_{p+1}, \cdots, x^*_n)$ is a basic solution of the complete canonical inequality (6-4).

(C) If $d > 0$ and $(x^*_2, \cdots, x^*_n)$ is a basic solution of

$$\sum_{j=2}^{n} C_j x_j \geq d - C_1,$$

then $(1, x^*_2, x^*_3, \cdots, x^*_n)$ is a basic solution of (6-4).

Result (A) allows partial paths to be excluded from further consideration when they are "dead-ended" by a PIN which has no solution. (Use the contrapositive form of statement (A).)

Repeated applications of (B) and (C) allow construction of complete BSP's from PIN basic solutions and PPR's. By repeatedly applying (B) and (C), one starts with a basic solution of the PIN and constructs a complete BSP by prefixing one element of the partial path record at a time to this basic solution. Results (B) and (C) validate the formation of a complete BSP by simply prefixing the PPR to the basic solution of a PIN at the node being visited.

6.52 Node Visits Summarized in Terms of PIN Parameters

By using (A),(B) and (C) of 6.51 above, Hammer and Rudeanu[87, 88,89] have built up the clever Solution Decision Table shown on Fig. 6-5. This table is important because it permits inferences to be made about the solutions of a PIN simply by inspection of its coefficients and right hand side.

The flow chart on Fig. 6-6 presents a modified version of this decision table which shows the sequence of calculations which are performed on the parameters of the PIN associated with the current node. This flow chart is applied when the node is 'visited'. Examining the parameters leads to a classification of the PIN into one of 7 mutually exclusive cases. Each of the 7 cases gives information about basic solutions and exclusion of neighboring nodes in the tree.

Thus at any node of the solution tree p basic solutions to the PIN may be identified where $p \leq n$. In addition, one or both of the branches extending from the current node may be excluded from further consideration.

Fig. 6-6 defines exactly what is done when a node is visited. This completes the discussion of this part of the TPA. Scheduling of node visits is next considered.

6.6 Scheduling Node Visits in the Binary Solution Tree

This section develops methods for the following items:

(a) scheduling node visits in the binary solution tree;

(b) maintenance of PPR's corresponding to the node being visited;

FIGURE 6-5

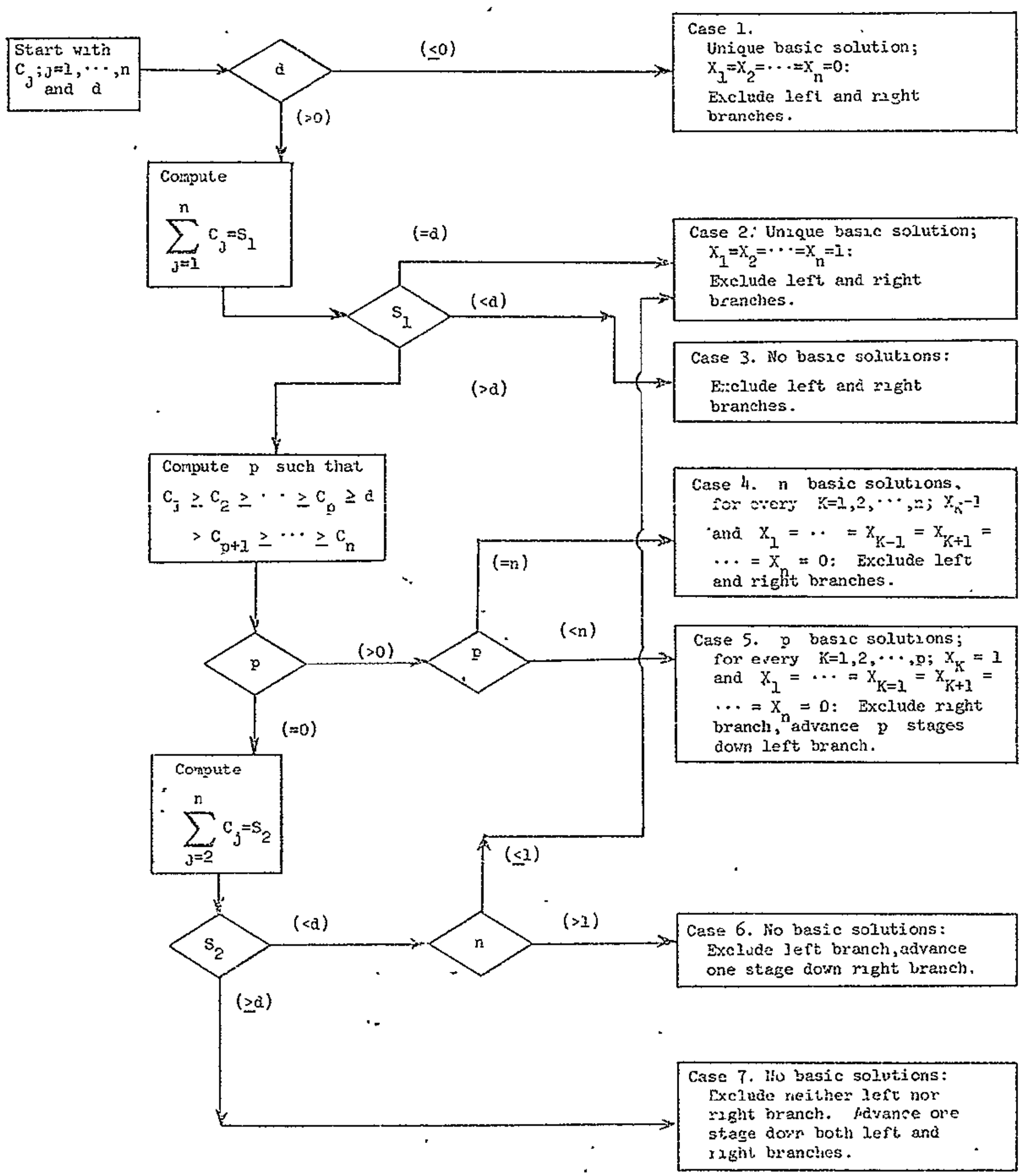
SOLUTION DECISION TABLE[1]

Case	Conclusions	Validation
$d \leqslant 0$	The unique basic solution is $x_1 = x_2 = \ldots = x_n = 0$	Obviously
$d > 0$ and $c_1 \geqslant \ldots \geqslant c_p \geqslant d > c_{p+1} \geqslant \ldots \geqslant c_n$	α) For every $k = 1,2,\ldots,p$: $x_k=1, x_1=\ldots=x_{k-1}=x_{k+1}=\ldots=x_n=0$ is a basic solution. β) The other basic solutions (if any) are characterized by the property: $x_1=\ldots=x_p = 0$, and $(x_{p+1},\ldots,x_n)$ is a basic solution of $\sum_{j=p+1}^{n} c_j x_j \geqslant d$	Obviously by (A) and (B)
$d>0, c_i<d (i=1,2,\ldots,n)$ and $\sum_{i=1}^{n} c_i<d$	No solutions	Obviously
$d>0, c_i<d (i=1,2,\ldots,n)$ and $\sum_{i=1}^{n} c_i = d$	The unique basic solution is $x_1 = x_2 = \ldots = x_n = 1$	Obviously
$d>0, c_i<d (i=1,2,\ldots,n)$ $\sum_{i=1}^{n} c_i \geqslant d$ and $\sum_{j=2}^{n} c_j<d$	The basic solutions (if any) are characterized by the property: $x_1 = 1$, and $(x_2,\ldots,x_n)$ is a basic solution of $\sum_{j=2}^{n} c_j x_j \geqslant d - c_1$	by (A) and (C)
$d>0, c_i<d (i=1,2,\ldots,n)$ $\sum_{i=1}^{n} c_i > d$ and $\sum_{j=2}^{n} c_j \geqslant d$	The basic solutions (if any) are characterized by the property: either $x_1 = 1$ and $(x_2,\ldots,x_n)$ is a basic solution of $\sum_{j=2}^{n} c_j x_j \geqslant d - c_1$ or: $x_1=0$ and $(x_2,\ldots,x_n)$ is a basic solution of $\sum_{j=2}^{n} c_j x_j \geqslant d$	by (A), (B), and (C)

[1]From: Peter L. Hammer and Sergiu Rudeanu, Pseudo-Boolean Methods for Bivalent Programming, Lecture Notes in Mathematics, Vol. 23, (Berlin, Heidelberg, New York: Springer-Verlag, 1966), page 27.

FIGURE 6-6

FLOW CHART SHOWING THE MODIFIED SOLUTION DECISION TABLE FOR CLASSIFYING PARTIAL INEQUALITIES



(c) maintenance of a PIN coefficients list corresponding to the node being visited.

Item (a) above is developed by first considering a simple algorithm for scheduling pre-order binary tree traversal. (Tree <u>traversal</u> is the process of visiting all nodes in some specified order[90]). This simple algorithm is presented in section 6.62. It does not allow the outcome of node visits to alter the schedule of other node visits. The entire tree must be defined prior to traversal in this simple algorithm.

Section 6.63 discusses modifications to the tree traversal algorithm (TTA) to permit tree pruning. Tree pruning is the process whereby the tree traversal schedule is modified by results obtained when tree nodes are visited.

Finally section 6.64 gives details on how the dynamic PPR and PIN records are maintained during the traversal.

Section 6.61 precedes all the above with a simple example of how the TPA should work to illustrate the problem of dynamic scheduling of node visits.

6.61 A Simple TPA Example Problem

Consider the tree shown in Fig. 6-2A. One method of starting at the root node and sequentially visiting each node in the tree only once is called <u>pre-order tree traversal</u>[91].

The pre-order traversal sequence applied to the tree gives the following order for node enumeration: $r \to a \to c \to g \to h \to d \to i \to j \to b \to e \to k \to l \to f \to m \to n$.

At each node in the order given above, the PIN is classified using Fig. 6-6.

For node r, we have case 6 of Fig. 6-6 since $d > 0$; $C_i < d$; $\sum_{i=1}^{n} C_i > d$; and $\sum_{i=2}^{n} C_i < d$. The basic solutions, if any, are found by setting $x_1 = 1$ and advancing one stage down to the right to node b. <u>We have bypassed the entire left branch of the tree</u> (where $x_1 = 0$). Thus we have eliminated nodes (a,c,d,g,h,i,j) from further consideration. This is an illustration of the pruning operation.

The revised schedule for pre-order traversal of the remainder of the tree is b→e→k→l→f→m→n. At node b we consider the PIN: $2x_1 + x_3 \geq 1$. This inequality matches case 4 of Fig. 6-6, since $C_1 > C_2 = d \Longrightarrow p = 2 = n$. Thus the basic solutions of the PIN are given by (1,0) and (0,1). Since $(x_1,x_2,x_3) = (1,-,-)$ is the PPR at node b, the BSP's to the original inequality are given by (1,1,0) and (1,0,1). This concludes the traversal process since all other nodes have been excluded, and the algorithm terminates after node b has been visited.

Thus by analyzing PIN's at two nodes of the 15-node tree, all the basic BSP's have been found. The ideas presented in this example represent the basic procedure used to identify all the BSP's in a solution tree.

6.62 The Pre-Order Tree Traversal Algorithm (TTA)

The general LPBI solution procedure has been illustrated in the preceding section. An important characteristic of this procedure is the successive re-definition of the traversal schedule which occurs as a result of node visits. A separate sub-algorithm to handle dynamic changes in the traversal schedule is needed.

The algorithm for dynamic scheduling used in the final TPA has been derived from a simpler algorithm called the pre-order TTA. The TTA allows no dynamic modification of the tree structure and requires that the entire tree be defined before node visiting begins. To promote understanding of the final TPA, the simpler TTA is presented here in detail.

There are three principal ways to traverse a binary tree, visiting each node once and only once. These methods all give rise to a specific ranking of the tree nodes in the order in which they will all be visited. They are termed pre-order, post-order and end-order traversal[92]. Pre-order traversal will be used here. It is defined by the following successive steps:

(a) visit the root;

(b) traverse the left subtree;

(c) traverse the right subtree. In the example stated previously in section 6.61, the tree of Fig. 6-2A has a pre-order traversal schedule given by: (r,a,c,g,h,d,i,j,b,e,k,l,f,m,n).

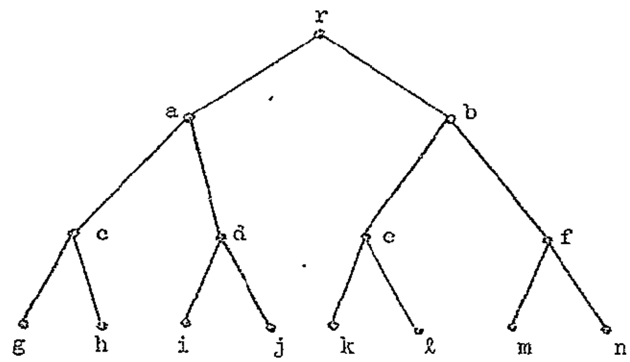
Before describing the method used to guarantee pre-order traversal, it is convenient to discuss three data structures required, namely a link table, a pushdown list and a single working storage location. The link table is necessary to show how the tree nodes are linked to each other. For the tree of Fig. 6-2A we can show a link diagram and corresponding link table (see Fig. 6-7). The tree structure is completely defined by the link table. Each tree node has a left and a right link to other nodes. Tree nodes are given an integer tag for internal machine use, but this tag can be related to other symbols via a look-up table. The null link is represented here by -1. The data structure STACK is a push-down, pop-up list with last-in, first-out (LIFO) discipline. STACK functions as a 'memory' for nodes remaining to be visited. A single storage location labeled P is also required to define the node currently being visited.

The following conventions will be used to describe data storage and data movement instructions. We read P←LLINK(P) as "replace the contents of memory location P with the contents of the memory location LLINK(P)". Memory location LLINK(P) is not modified by the preceding operation. For push-down list operations, we read P←STACK as "replace contents of memory location P with the contents of whichever memory location is at the top of the push-down list STACK". After this operation, the list STACK is to be popped up, or shortened by one item. The data transferred to P is no longer stored in STACK after the list is popped up. The list operation STACK←P means that "the contents of memory location P are to become the first item in the list STACK, on top

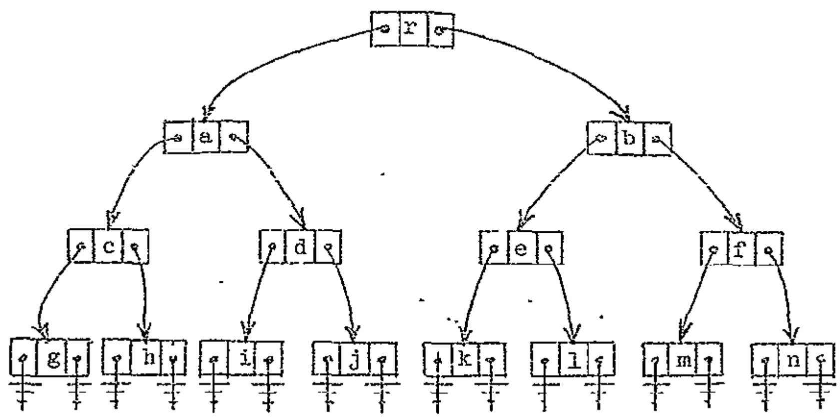
FIGURE 6-7

BINARY TREE WITH ASSOCIATED LINK DIAGRAM AND LINK TABLE

(A) Tree Diagram



(B) Link Diagram



(C) Link Table

Node	P	LLINK(P)	RLINK(P)
r	1	2	3
a	2	4	5
b	3	6	7
c	4	8	9
d	5	10	11
e	6	12	13
f	7	14	15
g	8	-1	-1
h	9	-1	-1
i	10	-1	-1
j	11	-1	-1
k	12	-1	-1
l	13	-1	-1
m	14	-1	-1
n	15	-1	-1

of elements already in STACK." This pushes down the list by adding one more element. The contents of memory location P are not modified.

The TTA is described by the flow chart of Fig. 6-8. (This description is similar to that for a post-order TTA given by Knuth[93].) Operations shown in this flow chart are numbered. Written descriptions of these operations are given below. These descriptions are numbered to correspond to the numbers of Fig. 6-8.

(1) P←ROOT. The number of the current node is replaced by the number of the root node. This is an initialization step. STACK is assumed empty.

(4) VISIT P. Some operation is performed at node P (such as investigating the parameters of an inequality).

(5) STACK←P. The node number in P is put on the push-down list STACK. (Note that the contents of P are not modified.)

(2) P←LLINK(P). The node number in P is replaced by the node number in LLINK(P) which is defined in the link table. This prepares for a move down the tree and to the left.

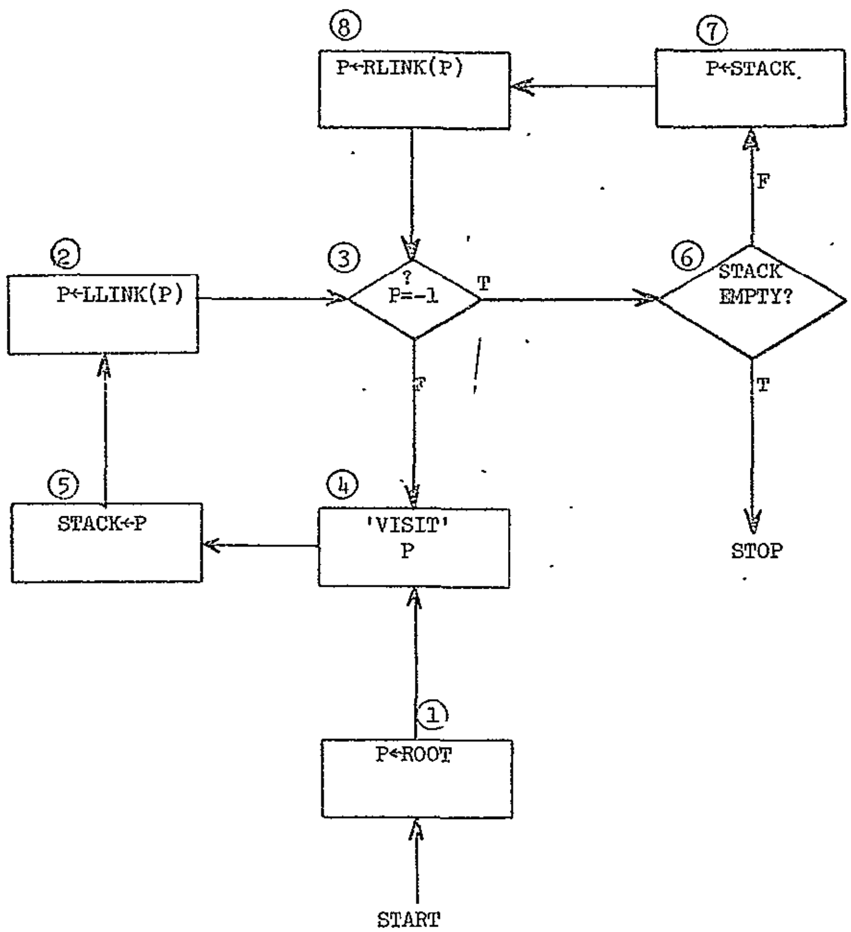
(3) P = -1?. Test to see if the contents of P are the null link. If yes, go to step (6) to determine whether STACK is empty. If no, go to step (4).

(6) STACK EMPTY?. If the push-down list STACK is empty, the algorithm is terminated. If STACK is not empty, go to step (7).

(7) P←STACK. Replace the contents of P with the node number at the top of the push-down list STACK. This pops up the list. The tree move is upward and to the right, back to the pivot node.

FIGURE 6-8

FLOWCHART SHOWING AN ALGORITHM FOR PRE-ORDER TRAVERSAL OF BINARY TREES



Note: The tree is assumed defined by a complete link table, with -1 as null link

Note: Algorithm steps are numbered to correspond to the descriptions given in the text

(8) P←RLINK(P). Replace the contents of P with the right link number of the current link in P. The tree move is downward and to the right, away from the pivot node.

Operation of the TTA can be further illustrated with an example using the tree of Fig. 6-2A. Fig. 6-9 shows a "snapshot" of the contents of the various memory locations after each step of the algorithm (as shown on Fig. 6-8) is completed. Thirty-nine sequential steps are shown, which caused nodes r,a,c,g,h,d to be visited in that order. Traversal of the rest of the tree in pre-order can be continued in the same way until node n has been explored, at which time the algorithm terminates.

The pre-order TTA consists of 2 types of operations:

(a) moving downward and to the left in the tree, one node at a time while retaining a record of downward moves (node numbers) in the push-down list; and

(b) moving back <u>up</u> to the right, one node at a time, by popping up the pushdown list, then moving down to the right, one stage. This is a 'back up and go round the corner' type of move.

Note that the push-down list STACK never contains more than (n+1) elements, where n is the number of levels (stages) in the tree.

6.63 Modifying the TTA to Permit Tree Pruning

The pre-order TTA presented in section 6.62 provides the basic framework for the TPA. However, there are two modifications of the TTA which must be made. These are discussed below.

FIGURE 6-9

EXAMPLE PROBELM ILLUSTRATING THE TREE PRE-ORDER TRAVERSAL ALGORITHM

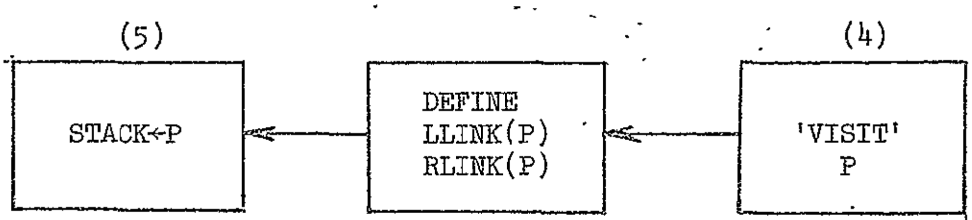
n	Step	P	Stack	Comments
1	1	1(r)	-------------------	INITIALIZE
2	4	1(r)	-------------------	VISIT P
3	5	1(r)	1(r)	STACK←P
4	2	2(a)	1(r)	P←LLINK(P)
5	3	2(a)	1(r)	TEST P
6	4	2(a)	1(r)	VISIT P
7	5	2(a)	2(a),1(r)	STACK←P
8	2	4(c)	2(a),1(r)	P←LLINK(P)
9	3	4(c)	2(a),1(r)	TEST P
10	4	4(c)	2(a),1(r)	VISIT P
11	5	4(c)	4(c),2(a),1(r)	STACK←P
12	2	8(g)	4(c),2(a),1(r)	P←LLINK(P)
13	3	8(g)	4(c),2(a),1(r)	TEST P
14	4	8(g)	4(c),2(a),1(r)	VISIT P
15	5	8(g)	8(g),4(c),2(a),1(r)	STACK←P
16	2	-1	8(g),4(c),2(a),1(r)	P←LLINK(P)
17	3	-1	8(g),4(c),2(a),1(r)	TEST P
18	6	-1	8(g),4(c),2(a),1(r)	TEST STACK
19	7	8(g)	4(c),2(a),1(r)	P←STACK
20	8	-1	4(c),2(a),1(r)	P←RLINK(P)
21	3	-1	4(c),2(a),1(r)	TEST P
22	6	-1	4(c),2(a),1(r)	TEST STACK
23	7	4(c)	2(a),1(r)	P←STACK
24	8	9(h)	2(a),1(r)	P←RLINK(P)
25	3	9(h)	2(a),1(r)	TEST P
26	4	9(h)	2(a),1(r)	VISIT P
27	5	9(h)	9(h),2(a),1(r)	STACK←P
28	2	-1	9(h),2(a),1(r)	P←LLINK(P)
29	3	-1	9(h),2(a),1(r)	TEST P
30	6	-1	9(h),2(a),1(r)	TEST STACK
31	7	9(h)	2(a),1(r)	P←STACK
32	8	-1	2(a),1(r)	P←RLINK(P)
33	3	-1	2(a),1(r)	TEST P
34	6	-1	2(a),1(r)	TEST STACK
35	7	2(a)	1(r)	P←STACK
36	8	5(d)	1(r)	P←RLINK(P)
37	3	5(d)	1(r)	TEST P
38	4	5(d)	1(r)	VISIT P
39	5	5(d)	5(d),1(r)	STACK←P
.	.	.	. . .	. . . .

LINK TABLE

Node	P	LLINK(P)	RLINK(P)
r	1	2	3
a	2	4	5
b	3	6	7
c	4	8	9
d	5	10	11
e	6	12	13
f	7	14	15
g	8	-1	-1
h	9	-1	-1
i	10	-1	-1
j	11	-1	-1
k	12	-1	-1
l	13	-1	-1
m	14	-1	-1
n	15	-1	-1

6.631 Elimination of the Pre-Determined Link Table. The traversal algorithm requires that the link table be defined before traversal. In the search for BSP's, many of the tree nodes will never be visited, since they will have been excluded (pruned away) from further consideration by results obtained at nodes nearer the root of the tree. Link table information for nodes to be excluded is not needed. To avoid defining the tree completely ahead of time, the tree is constructed by the algorithm itself, and the only nodes which are defined in the link table are those which must be visited, i.e. those which have not been pruned away by previous results obtained higher up in the tree. Thus the structure of the tree is actually determined as it is traversed.

Necessary modifications to the algorithm shown on Fig. 6-8 involve only the insertion of a new operation between the blocks labeled (4) and (5) as shown below:



This new operation is the definition of left and right links of node P. It can be considered as part of block (4) ('VISIT' P) if desired.

6.632 Storage Allocation Modifications. Defining a link table as the tree is traversed introduces practical considerations. How can identification numbers be assigned to new nodes? And, how much storage space is required for a link table used with a tree of given size?

One obvious method for assigning node numbers is to define a new sequential integer for each new node that is discovered. The size of the link table is then proportional to the size of the set of visited nodes, i.e.

$$1+2+4+\cdots+2^n = \sum_{k=0}^{n} 2^k = 2^{n+1} - 1$$

for a tree with <u>all nodes</u> visited. This is the maximum size of the link table and several values are shown below.

n	$\sum_{k=0}^{n} 2^k = 2^{n+1} - 1$
5	63
10	2047
15	65,535

Clearly this method is unworkable, since maximum storage space requirements are much too great.

The method used in the TPA is to use node numbers <u>over again</u>. The node number (index in the link table) is assigned to a <u>new node</u> once the node it originally was assigned to becomes <u>inactive</u>. From the description of the pre-order TTA, it can be seen that once a tree node is removed from the push-down list STACK (a 'back up and around the corner' move), this node is not utilized for any further processing and will be defined as being <u>inactive</u>. (<u>Active</u> nodes are defined as those nodes which are in the push-down list STACK, or those nodes which are <u>right links</u> of nodes in STACK, since right link nodes may become occupants of STACK.) Once nodes become inactive, their node numbers become eligible for re-assignment to new nodes.

Since the maximum number of nodes in STACK at one time is (n+1), and since each node has only one possible right link, the maximum number of active nodes will be 2(n+1). Thus, the dynamic link table will contain at most 2(n+1) node link records. Also only 2(n+1) unique node numbers will ever be needed at one time.

In order to assign node numbers as needed during traversal of the tree, a second push-down list PLIST is initially loaded with 2(n+1) consecutive integers so that the first integer removed is 1. As the tree is traversed, new nodes may be identified. These new nodes will be assigned numbers taken from the top of PLIST which pops up the list.

Numbers from inactive nodes are placed on the top of PLIST which pushes down the list. This occurs as soon as the nodes become inactive, or between steps (7) and (8) of Fig. 6-8 (between the 'move back up', and the 'move down right').

The TPA with the modifications necessary to provide for the dynamic link table is shown in Fig. 6-10.

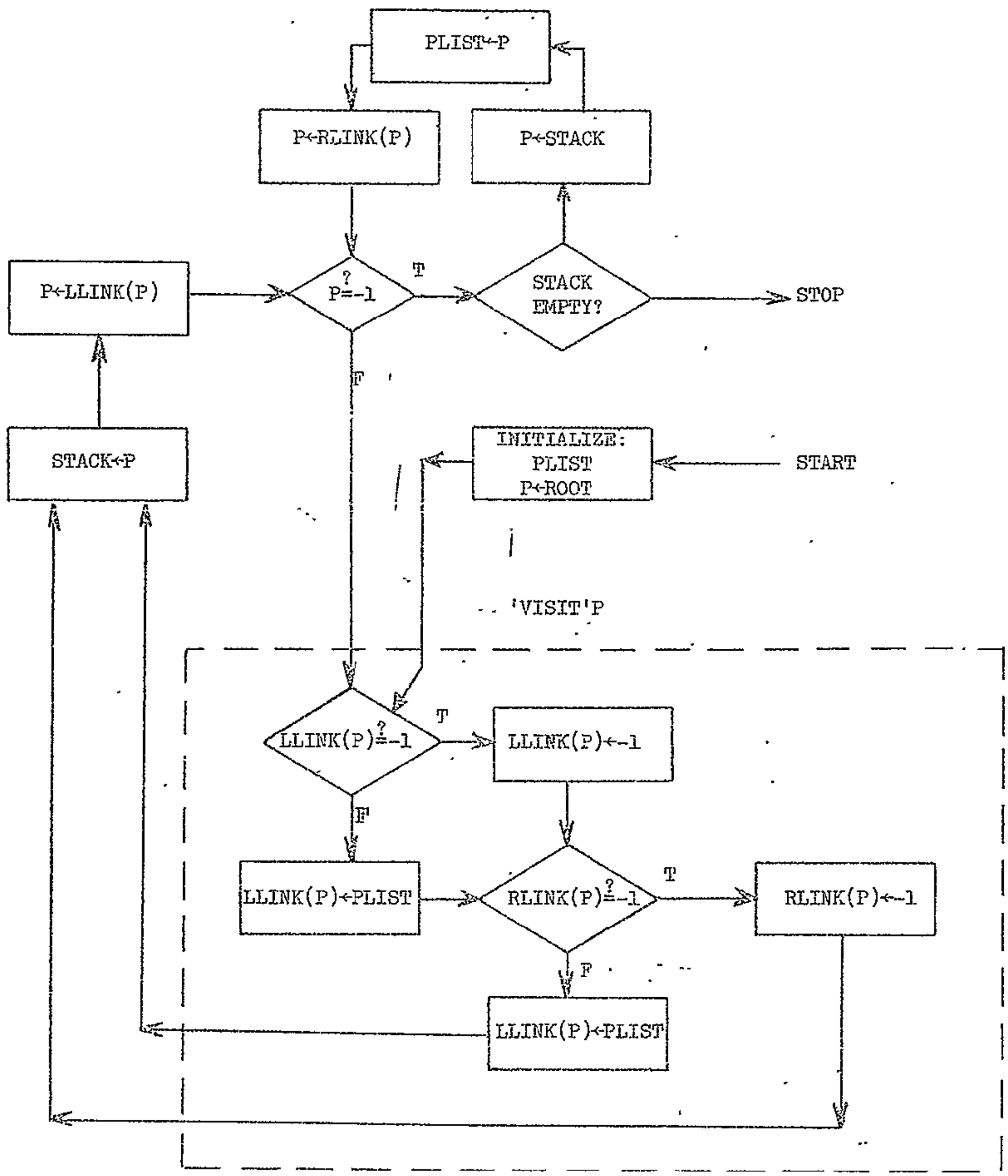
6.64 Maintaining the Dynamic PPR and PIN Records

6.641 Maintenance of the PPR. As was discussed in sections 6.41 and 6.51, a complete BSP of the canonical inequality is constructed from two components. The PPR from the root to node P is required together with a basic solution of the PIN associated with node P.

In addition, the PPR is required to form the right hand side of the PIN from the right hand side of the complete canonical inequality.

FIGURE 6-10

FLOWCHART OF THE TREE TRAVERSAL ALGORITHM AFTER MODIFICATION TO PERMIT GENERATION OF A DYNAMIC LINK TABLE



The above two uses require that the PPR be recorded and updated as the various tree nodes are sequentially visited. This represents an addition to the pre-order TTA.

The PPR is an ordered list of 0's and 1's (left and right branches) along the path from the root node to the current node P.

A running record of the partial path is kept in a binary vector Y(J) having n elements. The index of the last element of Y(J) which is recorded represents the 'level' in the tree where the current node P is located. Recall that the 'level' associated with any node P ranges from 0 (the root node) to n (the leaf nodes). This level is called STAGE(P) in the trees of Figs. 6-2, 6-3, and 6-4. The variable STAGE(P) is assigned as an attribute to each new node P in the dynamic link table at the same time LLINK(P) and RLINK(P) are defined. STAGE(P) is retained as part of the node record in the dynamic link table.

As the partial path grows downward and to the left, 0's are added to the list Y(J). As the partial path is retraced back up the tree and downward to the right, the list Y(J) is first shortened and then expanded with 1's reflecting the rightward move.

To permit the list Y(J) to be modified as the tree is traversed, two pointers PT1 and PT2 (called the following and lead pointers respectively) which refer to elements in Y(J) are used.

As movement proceeds <u>downward and to the left</u> in the tree, the pointers and the PPR are revised according to the following rules:

$$\begin{aligned} PT1 &\leftarrow PT2 \\ PT2 &\leftarrow STAGE(P) \\ Y(J) &\leftarrow 0;\ J=PT1+1,\cdots,PT2. \end{aligned} \tag{6-20}$$

These operations expand the list $Y(J)$ by adding zeros.

Fig. 6-11 illustrates how the PPR is dynamically modified. Suppose the partial path and PPR shown in Fig. 6-11A exist at some time during enumeration of the tree. This is to be regarded as initial data. Next, suppose a move is made extending the initial partial path down two stages to the left. Revised data is given in Fig. 6-11B, after using (6-20).

As movement proceeds <u>back up the tree and then downward and to the right</u>, the pointers and the PPR are revised according to the following rules:

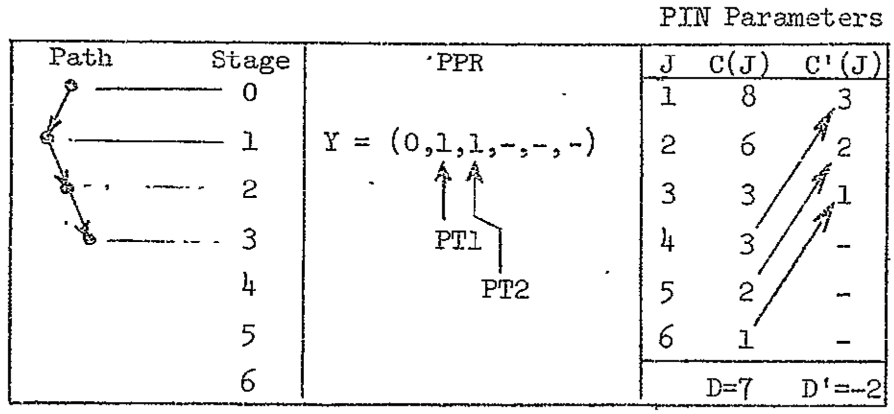
$$\begin{aligned} P &\leftarrow STACK \\ PT1 &\leftarrow STAGE(P) \\ P &\leftarrow RLINK(P) \\ PT2 &\leftarrow STAGE(P) \\ Y(J) &\leftarrow 1;\ J=PT1+1,\cdots,PT2\ . \end{aligned} \tag{6-21}$$

For example, starting with the data shown in Fig. 6-11B, assume a move is made back up the tree and down to the right. The final results are shown in Fig. 6-11C. The PPR $Y(J) = 0$ is erased as movement proceeds back up the tree, and overwritten with $Y(J) = 1$ as movement proceeds down and to the right.

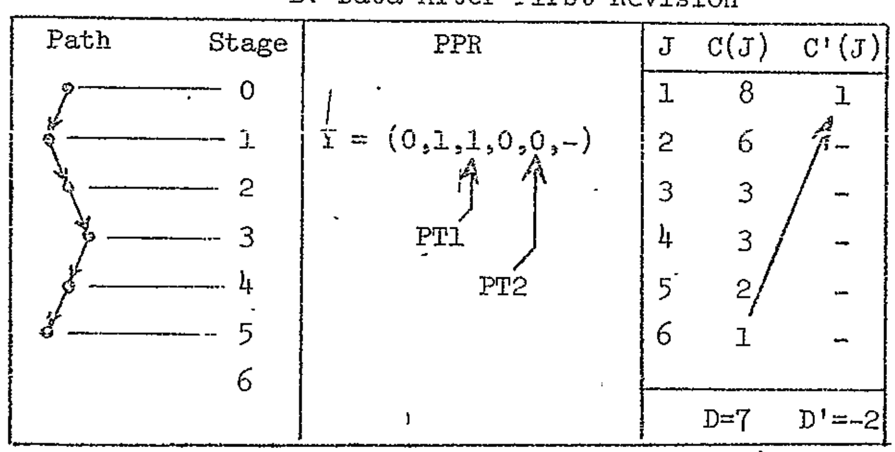
FIGURE 6-11

DETAILS OF PPR AND PIN DYNAMIC MODIFICATIONS

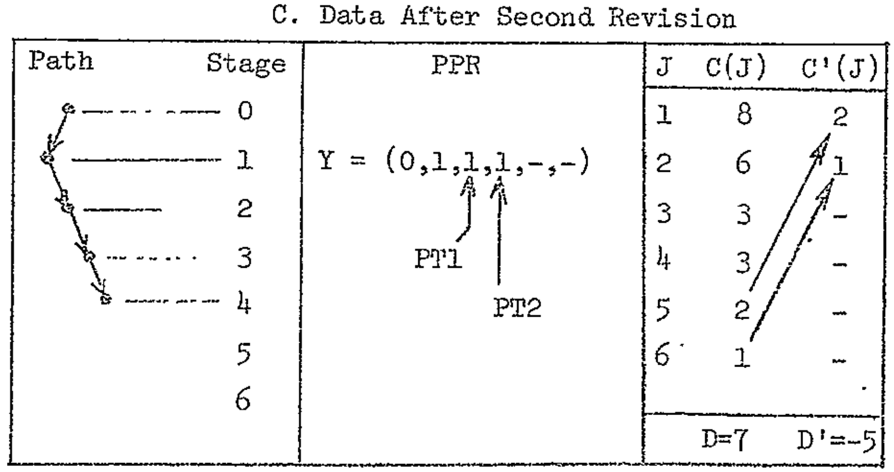
A. Initial Data



B. Data After First Revision



C. Data After Second Revision



6.642 Maintenance of PIN Coefficients. Provisions are also made for dynamically updating coefficients and right hand side of the PIN as the tree is traversed. This is another addition to the TTA.

A list of coefficients $C'(J)$, $J=1,2,\cdots M$ is maintained by using the list of coefficients $C(J)$, $J=1,2,\cdots,N$ for the original inequality. The $C'(J)$ are copied from the list of $C(J)$ as follows:

$$\begin{aligned} M &\leftarrow N - PT2 \\ K &\leftarrow PT2 + L \\ C'(L) &\leftarrow C(K);\ L=1,2,\ldots,M\ . \end{aligned} \tag{6-22}$$

The right hand side D' of the current PIN is then given in terms of the original right hand side D as:

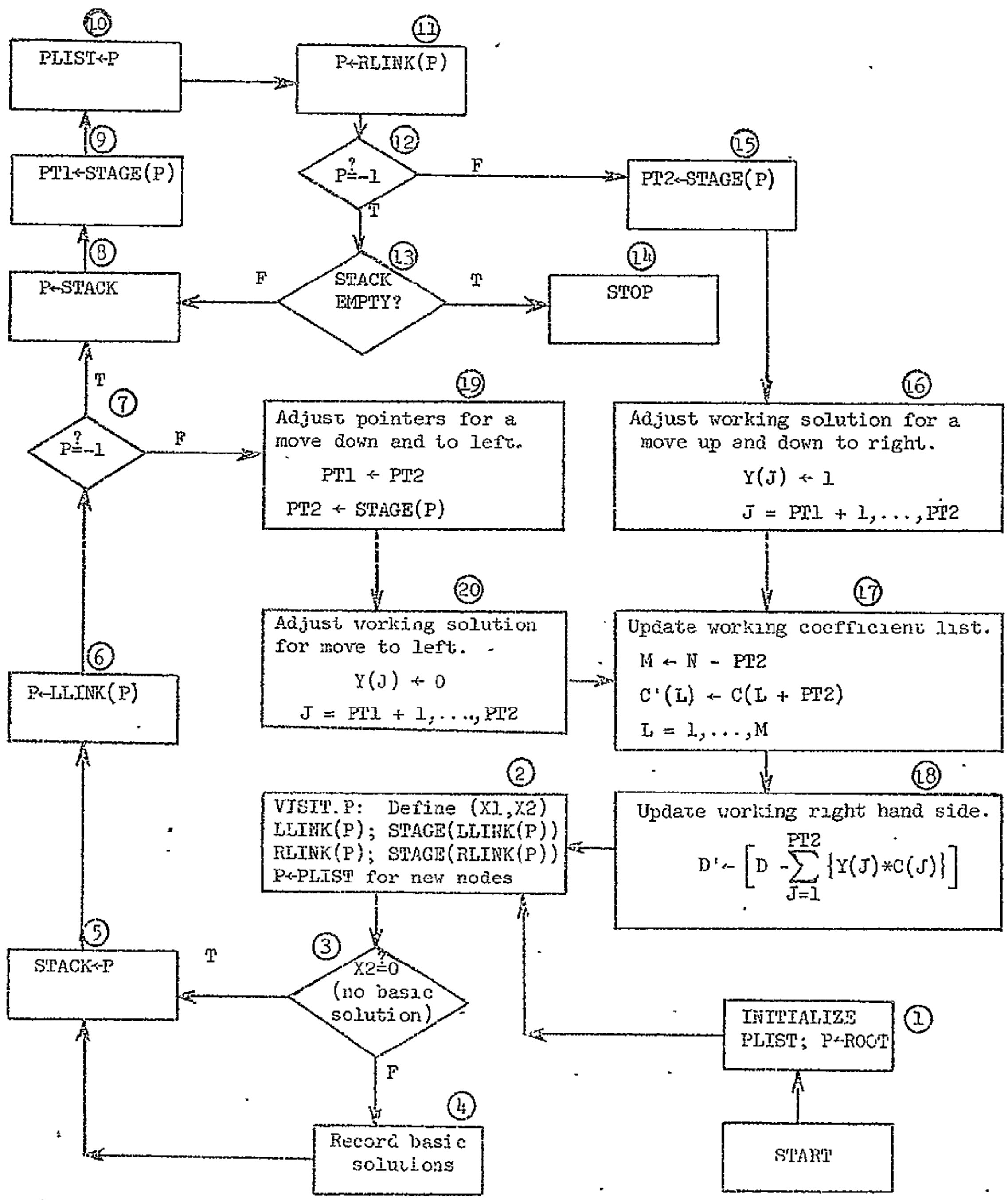
$$D' \leftarrow D - \sum_{J=1}^{PT2} [C(J)*Y(J)]. \tag{6-23}$$

An example of PIN parameter revision using (6-20) and (6-22) is shown in Fig. 6-11B. Fig. 6-11C uses (6-21), (6-22) and (6-23).

This completes the discussion of modifications and additions necessary to convert the pre-order TTA to the TPA. A flow chart of the TPA is shown in Fig. 6-12. A detailed description of this flow chart is presented in the next section.

FIGURE 6-12

FLOW CHART OF THE TREE PRUNING ALGORITHM FOR SOLVING PSEUDO-BOOLEAN INEQUALITIES IN CANONICAL FORM



Note: Algorithm steps are numbered to correspond to the descriptions given in the text.

6.7 The Tree-Pruning Algorithm (TPA)

6.71 Detailed TPA Description

The TPA is described by the flow chart of Fig. 6-12. The various operations shown in this flow chart are numbered and the written descriptions given below refer to these numbered operations.

(1) Initialize. The push-down list of new node numbers PLIST is loaded with sequential integers $1,2,\cdots,2(N+1)$. The first integer (unity) is removed from PLIST and placed in P to correspond to the root node. STAGE(P) ← 0 for the root node.

The PPR is undefined at the root node. The pointers PT1 and PT2 are both set to zero.

The PIN parameters C'(J) and D' are set equal to the canonical inequality parameters C(J) and D.

(2) Visit P. The PIN associated with node P is classified using Fig. 6-6. X1 is the classification case number and X2 is the number of PIN basic solutions identified. If nodes linked to P are identified, they are assigned numbers from the push-down list PLIST. These node numbers are entered in the dynamic link table as LLINK(P), RLINK(P) or both. They appear as part of the node P record. Also each node linked to P has its attribute STAGE(LLINK(P)), STAGE(RLINK(P)), or both recorded in the dynamic link table at this time.

(3) Test for basic solutions. If X2=0, then no PIN basic solutions were identified when P was visited.

(4) Construct and record all complete BSP's discovered. All PIN basic solutions (X2 of them) identified in step (2) are used here. Each complete BSP is constructed using the PPR and a PIN basic solution. The form of each PIN basic solution is determined indirectly by X1 from step (2).

(5) STACK←P. The number of the node just visited is placed on the top of the push-down list STACK.

(6) P←LLINK(P). The number of the node just visited in step (2) is replaced by the number of its left link node. The movement is downward and to the left in the tree.

(7) Test for null link. Here the test $P \stackrel{?}{=} -1$ is performed to determine whether the node visited in step (2) has a left link to a new node. If no link node exists downward to the left, then control transfers to step (8) for a move back up the tree to the node visited in step (2). This is followed by a move down and to the right. If a left link does exist to a node further down the tree, then control transfers to step (19) for updating the PPR and the PIN parameters.

(8) P←STACK. The number of the node visited last is removed from the push-down list. This node is the pivot node for a move around the corner and down to the right.

(9) PT1←STAGE(P). The following pointer in the PPR is moved back to the stage of the pivot node (sometimes this step results in no actual movement of the pointer).

(10) PLIST←P. Since the pivot node will not be needed again, it becomes inactive and its node number is released for future use by

new nodes. This is done by placing the node number back on the push-down list PLIST.

(11) P←RLINK(P). The number of the pivot node P is replaced by the number of its right link for a move down the tree and to the right.

(12) Test for null link. Here we test whether the right link of the pivot is non-null. If it <u>is null</u>, then go to step (13) to test for an empty STACK. If it <u>is not null</u>, then go to step (15) to prepare to visit the node.

(13) Test for empty STACK. If the push-down list STACK is empty, then the algorithm is terminated at step (14) and the tree has been completely traversed.

(14) STOP. The tree has been traversed.

(15) PT2 ← STAGE(P). The <u>lead</u> PPR pointer is moved ahead to correspond to the move down the tree. (STAGE(P) was established in step (2)).

(16) Y(J) ← 1; J=PT1+1,···,PT2. The PPR is expanded to reflect the move down the tree and to the right.

(17) M ← N - PT2; C'(L) ← C(L+PT2), L=1,...,M. The PIN coefficients are updated to correspond to the node P which will be visited.

(18) $D' \leftarrow D - \sum_{J=1}^{PT2} [Y(J) \times C(J)]$. The right hand side is adjusted to correspond to the PIN associated with the node P which will be visited.

(19) PT1 ← PT2;PT2 ← STAGE(P). Advance both leading and following PPR pointers to correspond to a move down the tree and to the left.

(20) Y(J) ← 0;J=PT1+1···,PT2. The PPR is expanded to reflect the move down the tree and to the right.

6.72 Example Problems

Two example problems are given here. First, the very simple example used in section 6.61 and shown in Fig. 6-2 is presented here in detail. This example shows step-by-step operation of the TPA. It is discussed in 6.721 below.

The second example (in section 6.722 below) illustrates the entire LPBI solution process. This includes:

(a) illustration of the parameter transformation to canonical form;

(b) an overview of basic solution determination using the TPA;

(c) generation of canonical solution families from basic solutions;

(d) transformation of canonical solution families to general solution families.

The LPBI used in the second example is the same one discussed in section 6.352 and illustrated in Figures 6-3 and 6-4.

6.721 <u>A Detailed Example of the TPA</u>. Fig. 6-2A shows the complete solution tree for the inequality

$$3x_1 + 2x_2 + x_3 \geq 4$$

By applying the TPA of Fig. 6-12 to this tree, we can solve the inequality. The general method of doing this was illustrated in section 6.61. Fig. 6-13 shows the detailed results as the TPA of Fig. 6-12 is applied. Each step of Fig. 6-13 corresponds to a numbered block in the flow chart of Fig. 6-12. The status of all data structures except the link table is shown in Fig. 6-13. The status of the dynamic link table is illustrated in Fig. 6-14 as it is modified during the tree traversal. Only two records appear in this link table because only two tree nodes are visited before all solutions are found.

6.722 Solving the General Form Inequality. This example follows the solution of the inequality

$$-2z_1 - 3\bar{z}_2 + 5\bar{z}_3 - z_4 + 2z_5 \geq 0.$$

The solution tree to this general form inequality is illustrated in Fig. 6-3. The transformation of the inequality parameters to canonical form is shown in Fig. 6-15A. The same transformations were used as an example in section 6.321. They are presented again in 6-15A with other transformations required for the complete solution of this inequality. The solution tree associated with this canonical inequality is shown on Fig. 6-4.

The application of the TPA of Fig. 6-12 is illustrated below to find the seven basic solutions indicated in Fig. 6-15B and in Fig. 6-4. Node visits are presented sequentially and detailed results are shown. Each paragraph below corresponds to a single node visit. The growth of

FIGURE 6-13

EXAMPLE PROBLEM SHOWING DETAILS OF THE TREE PRUNING ALGORITHM FOR THE INEQUALITY $3X_1 + 2X_2 + X_3 \geq 4$

n	*Step	P	PLIST	Y(J)	STACK	PT1	PT2	DATA	RHS	NCOEF	X1	X2	Comments
1	1	1(r)	2,3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	-	-	INITIALIZE
2	2	1(r)	3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	VISIT P ; MODIFY LINK TABLE
3	3	1(r)	3,4,5,6,7	(-,-,-)	1(r)	0	0	3,2,1	4	3	6	0	TEST X2
4	5	1(r)	3,4,5,6,7	(-,-,-)	1(r)	0	0	3,2,1	4	3	6	0	STACK←P
5	6	-1	3,4,5,6,7	(-,-,-)	1(r)	0	0	3,2,1	4	3	6	0	P←LLINK(P)
6	7	-1	3,4,5,6,7	(-,-,-)	1(r)	0	0	3,2,1	4	3	6	0	TEST P
7	8	1(r)	3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	P←STACK
8	9	1(r)	3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	PT1←STAGE(P)
9	10	1(r)	1,3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	PLIST←P
10	11	2(b)	1,3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	P←RLINK(P)
11	12	2(b)	1,3,4,5,6,7	(-,-,-)	----	0	0	3,2,1	4	3	6	0	TEST P
12	15	2(b)	1,3,4,5,6,7	(-,-,-)	----	0	1	3,2,1	4	3	6	0	PT2←STAGE(P)
13	16	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	3,2,1	4	3	6	0	ADJUST Y(J)
14	17	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	2,1	4	2	6	0	ADJUST COEF. LIST
15	18	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	2,1	1	2	6	0	ADJUST RIGHT HAND SIDE
16	2	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	2,1	1	2	2	2	VISIT P ; MODIFY LINK TABLE
17	3	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	2,1	1	2	2	2	TEST X2
18	4	2(b)	1,3,4,5,6,7	(1,-,-)	2(b)	0	1	2,1	1	2	2	2	RECORD BASIC SOLUTIONS
19	5	2(b)	1,3,4,5,6,7	(1,-,-)	2(b)	0	1	2,1	1	2	2	2	STACK←P
20	6	-1	1,3,4,5,6,7	(1,-,-)	2(b)	0	1	2,1	1	2	2	2	P←LLINK(P)
21	7	-1	1,3,4,5,6,7	(1,-,-)	2(b)	0	1	2,1	1	2	2	2	TEST P
22	8	2(b)	1,3,4,5,6,7	(1,-,-)	----	0	1	2,1	1	2	2	2	P←STACK
23	9	2(b)	1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	PT1←STAGE(P)
24	10	2(b)	2,1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	PLIST←P
25	11	-1	2,1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	P←RLINK(P)
26	12	-1	2,1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	TEST P
27	13	-1	2,1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	TEST STACK
28	14	-1	2,1,3,4,5,6,7	(1,-,-)	----	1	1	2,1	1	2	2	2	STOP

*Step refers to algorithm step shown on flow chart and described in the text

FIGURE 6-14

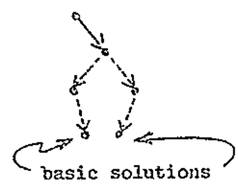
CONTINUATION OF EXAMPLE PROBLEM SHOWING DETAILS OF TREE PRUNING ALGORITHM

Original Data (Canonical Form)

J	Coef(J)
1	3
2	2
3	1

D=4

Exploration Path



DYNAMIC LINK TABLE CONSTRUCTION FOR EXAMPLE PROBLEM

n = 1 (INITIALIZATION)

J = P	LLINK(P)	RLINK(P)	STAGE(P)
1	--	--	0
2			
3			
4			
5			
6			
7			

n = 2 (VISIT ROOT)

J = P	LLINK(P)	RLINK(P)	STAGE(P)
1	-1	2	0
2	--	--	1
3			
4			
5			
6			
7			

$3x_1 + 2X_2 + X_3 \geq 4$

No basic solutions

n = 16 (VISIT NODE b)

J = P	LLINK(P)	RLINK(P)	STAGE(P)
1	-1	2	0
2	-1	-1	1
3			
4			
5			
6			
7			

$2X_1 + X_2 \geq 1$

Two basic solutions

(1,1,0)
(1,0,1)

FIGURE 6-15

EXAMPLE PROBLEM SHOWING TRANSFORMATIONS INVOLVED IN THE SOLUTION OF A LINEAR PSEUDO-BOOLEAN INEQUALITY

A. Parameter Transformation to Canonical Form

$$\Sigma z_j^{\alpha_j}\gamma_j \geq \delta \longrightarrow \Sigma y_j^1 e_j \geq d \longrightarrow \Sigma x_j^1 c_j \geq d$$

j	γ_j	α_j
1	-2	1
2	-3	0
3	5	0
4	-1	1
5	2	1
	$\delta=0$	

$\longrightarrow$

e_j	a_j
2	0
3	1
5	0
1	0
2	1
$d=6$	

$\longrightarrow$

c_j	$P(j)$
5	3
3	2
2	1
2	5
1	4
$d=6$	

B. Conversion of Basic Solutions to Canonical Solution Families

Basic Solutions

$\underline{x}_k$; $k=1,2,\cdots,7$

j	1	2	3	4	5	6	7
1	0	0	0	1	1	1	1
2	1	1	1	1	0	0	0
3	0	1	1	0	1	0	0
4	1	1	0	0	0	1	0
5	1	0	1	0	0	0	1

Canonical Solution Families

$F_k(\underline{x})$, $k=1,2,\cdots,7$

j	1	2	3	4	5	6	7
1	0	0	0	1	1	1	1
2	1	1	1	1	0	0	0
3	0	1	1	-	1	0	0
4	1	1	0	-	-	1	0
5	1	-	1	-	-	-	1

C. Transformation of Canonical Solution Families to Central Form Solution Families

Intermediate solution families

$F_k(\underline{y})$; $k=1,2,\cdots,7$

j	1	2	3	4	5	6	7
1	0	1	1	-	1	0	0
2	1	1	1	1	0	0	0
3	0	0	0	1	1	1	1
4	1	-	1	-	-	-	1
5	1	1	0	-	-	1	0

Transformation

$\ell \leftarrow P(j)$

$y_{k,\ell} \leftarrow x_{k,j}$

$j=1,\cdots,5$

$k=1,\cdots,7$

General solution families

$F_k(\underline{z})$; $k=1,2,\cdots,7$

j	1	2	3	4	5	6	7
1	1	0	0	-	0	1	1
2	1	1	1	1	0	0	0
3	1	1	1	0	0	0	0
4	0	-	0	-	-	-	0
5	1	1	0	-	-	1	0

Transformation

$a_j=0 \Rightarrow (z_{k,j} \leftarrow 1-y_{k,j})$

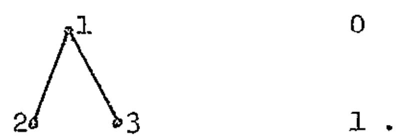
$a_j=1 \Rightarrow (z_{k,j} \leftarrow y_{k,j})$

$j=1,\cdots,5$

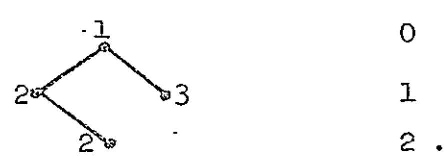
$k=1,\cdots,7$

the pruned subtree resulting from the node visit is also shown graphically.

(A) Visit node 1 (root). The node records give PIN = $5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 6$ and PPR = Y = (-,-,-,-,-). Node classification parameters are given by: $d > 0$; $s_1 = 13 > d$; $p = 0$; and $s_2 = 8 > d$. Node classification is case 7. There are no basic solutions and no exclusions. Advance one stage down both right and left branches. Define two new nodes. Label them 2 and 3. The tree is now defined as:

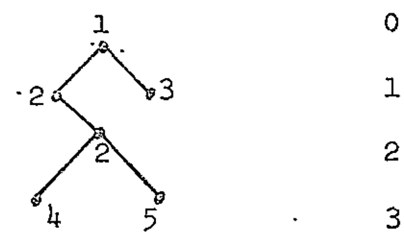


(B) Visit node 2 at stage 1. The node records give: PIN = $3x_2 + 2x_3 + 2x_4 + x_5 \geq 6$ and PPR = Y = (0,-,-,-,-). Node classification parameters are given by: $d > 0$; $s_1 = 8 > d$; $p = 0$; $s_2 = 5 < d$; and $n = 4 > 1$. Node classification is case 6. There are no basic solutions. Exclude the left branch, and advance one stage down the right branch. Define a new node. Label it 2, since the pivot node 2 has become inactive, and its number may be used over again. The tree is now defined as:

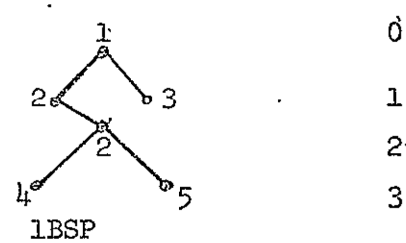


(C) Visit node 2 at stage 2. The node records give: PIN = $2x_1 + 2x_4 + x_5 \geq 3$ and PPR = Y = (0,1,-,-,-). Node classification parameters are given by: $d > 0$; $s_1 = 5 > d$; $p = 0$ and $s_2 = 3 \geq d$.

Node classification is case 7. There are no basic solutions. Exclude neither the right nor left branches. Advance one stage down both the left and right branches. Define two new nodes. Label them 4 and 5. The tree is now defined as:

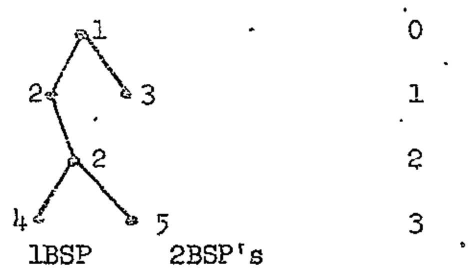


(D) Visit node 4 at stage 3. The node records give: PIN = $2x_4 + x_5 \geq 3$ and PPR = $\underline{Y}$ = (0,1,0,-,-). Node classification parameters are given by $d > 0$ and $s_1 = 3 = d$. Node classification is case 2. There is a unique basic solution. Exclude both left and right branches. Define no new nodes. The PIN basic solution is $(x_4,x_5) = (1,1)$ and the BSP is $\underline{x} = (0,1,0,1,1)$. The tree is now defined as:

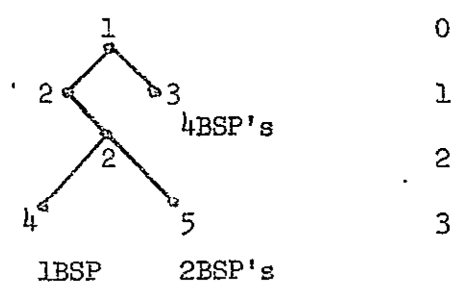


(E) Visit node 5 at stage 3. The node records give: PIN = $2x_4 + x_5 \geq 1$ and PPR = Y = (0,1,1,-,-). Node classification parameters are given by: $d > 0$; $s_1 = 3 > d$; and $p = 2 = n$. Node classification is case 4. There are two basic solutions. Exclude both left and right branches. Define no new nodes. The PIN basic solutions are:

$(x_4, x_5) = (1,0)$ and $(0,1)$. The BSP's are: $\underline{x} = (0,1,1,1,0)$ and $(0,1,1,0,1)$. The tree is now defined as:



(F) Visit node 3 at stage 1. The node records give: PIN = $3x_2 + 2x_3 + 2x_4 + x_5 \geq 1$ and PPR = Y $(1,-,-,-,-)$. Node classification parameters are given by: $d > 0$; $s_1 = 8 > d$; and $p = 4 = n$. Node classification is case 4. There are four basic solutions. Exclude both right and left branches. Define no new nodes. The PIN basic solutions are $(x_2, x_3, x_4, x_5) = (1,0,0,0)$ and $(0,1,0,0)$ and $(0,0,1,0)$ and $(0,0,0,1)$. The BSP's are: $\underline{x} = (1,1,0,0,0)$ and $(1,0,1,0,0)$ and $(1,0,0,1,0)$ and $(1,0,0,0,1)$. The tree is now defined as:



(G) The tree traversal ends. All nodes which were defined have been visited.

By visiting six nodes in a subtree (out of a possible 63 nodes in the complete tree), seven basic solutions were found.

Figure 6-4 shows that there are 19 binary solution vectors to the inequality. These solution vectors are clustered in families to the right of the basic solution path. There are an average of 19/7=2.7 solution vectors per family for this problem. Fig. 6-15B illustrates the conversion of the basic solutions to canonical solution families. All trailing 0's are changed to (-) to indicate arbitrary (0/1) variables. Fig. 6-15C shows the transformation of the canonical solution families back to the general form. This transformation takes place in two steps. First is the inverse permutation. Next, the complemented variables are accounted for.

6.73 Miscellaneous.

Fig. 6-16 is an enumeration of the transformation from canonical form solutions to general form solutions for the example problem of Fig. 6-4. In the left column of Fig. 6-16 are the 32 binary vectors $\underline{x}_k = (x^1_{k1}, \cdots x^1_{k5})$. In the right column are the corresponding transformed binary vectors $\underline{z}_k = (z^1_{k1}, \cdots, z^1_{k5})$. The families of solutions indicated on Fig. 6-15B and 6-15C are shown grouped in Fig. 6-16. Using Fig. 6-16, the following items can be noted.

$$\text{(A)} \quad u_k = \sum_{j=1}^{5} c_j x^1_{kj} = \sum_{j=1}^{5} z^{\alpha_j}_{kj} \gamma_j + g; \text{ where } g = -\sum_{(\gamma_j<0)} \gamma_j = 6.$$

This is an illustration of result (6-18). By (6-19) the transformation is order preserving.

FIGURE 6-16

EXAMPLE PROBLEM SHOWING ENUMERATION OF SOLUTION VECTOR BEFORE AND AFTER TRANSFORMATION

Combination number	Variable (X_j^1) 1	2	3	4	5	$\sum_{j=1} c_j X_j^1$		Variable (Z_j^1) 1	2	3	4	5	$\sum_{j=1}^{5} Z_j^{\alpha_j} \gamma_j$	
32	1	1	1	1	1	13*		0	1	0	0	1	7*	
31	1	1	1	1	0	12*		0	1	0	1	1	6*	
30	1	1	1	0	1	11*		0	1	0	0	0	5*	
29	1	1	1	0	0	10*	$F_4(\underline{X})$	0	1	0	1	0	4*	$F_4(\underline{Z})$
28	1	1	0	1	1	11*		1	1	0	0	1	5*	
27	1	1	0	1	0	10*		1	1	0	1	1	4*	
26	1	1	0	0	1	9*		1	1	0	0	0	3*	
25	1	1	0	0	0	8*		1	1	0	1	0	2*	
24	1	0	1	1	1	10*		0	0	0	0	1	4*	
23	1	0	1	1	0	9*	$F_5(X)$	0	0	0	1	1	3*	$F_5(\underline{Z})$
22	1	0	1	0	1	8*		0	0	0	0	0	2*	
21	1	0	1	0	0	7*		0	0	0	1	0	1*	
20	1	0	0	1	1	8*	$F_6(\underline{X})$	1	0	0	0	1	2*	$F_6(\underline{Z})$
19	1	0	0	1	0	7*		1	0	0	1	1	1*	
18	1	0	0	0	1	6*	$F_7(\underline{X})$	1	0	0	0	0	0*	$F_7(\underline{Z})$
17	1	0	0	0	0	5		1	0	0	1	0	-1	
16	0	1	1	1	1	8*	$F_2(\underline{X})$	0	1	1	0	1	2*	$F_2(\underline{Z})$
15	0	1	1	1	0	7*		0	1	1	1	1	1*	
14	0	1	1	0	1	6*	$F_3(\underline{X})$	0	1	1	0	0	0*	$F_3(\underline{Z})$
13	0	1	1	0	0	5		0	1	1	1	0	-1	
12	0	1	0	1	1	6*	$F_1(X)$	1	1	1	0	1	0*	$F_1(\underline{Z})$
11	0	1	0	1	0	5		1	1	1	1	1	-1	
10	0	1	0	0	1	4		1	1	1	0	0	-2	
9	0	1	0	0	0	3		1	1	1	1	0	-3	
8	0	0	1	1	1	5		0	0	1	0	1	-1	
7	0	0	1	1	0	4		0	0	1	1	1	-2	
6	0	0	1	0	1	3		0	0	1	0	0	-3	
5	0	0	1	0	0	2		0	0	1	1	0	-4	
4	0	0	0	1	1	3		1	0	1	0	1	-3	
3	0	0	0	1	0	2		1	0	1	1	1	-4	
2	0	0	0	0	1	1		1	0	1	0	0	-5	
1	0	0	0	0	0	0		1	0	1	1	0	-6	
$5X_1 + 3X_2 + 2X_3 + 2X_4 + X_5 \geq 6$								$-2Z_1 - 3\overline{Z}_2 + 5\overline{Z}_3 - Z_4 + 2Z_5 \geq 0$						

NOTE:

$$\sum_{j=1}^{5} X_j^1 c_j - \sum_{j=1}^{5} Z_j^{\alpha_j} \gamma_j = - \sum_{(\gamma_j<0)} \gamma_j = g = 2 + 3 + 1 = 6$$

(B) The canonical form vectors are shown in standard order, but note that

$$u_k = \sum_{j=1}^{5} c_j x_{kj}^1 \not\leq \sum_{j=1}^{5} c_j x_{\ell j} \qquad \text{for} \quad k \geq \ell,$$

which shows that the sequence u_k, $k=1,2,\cdots,32$ is <u>not</u> monotonically increasing. For example, $u_{29} < u_{28}$. Note also that $u_{12} = 6$, so $\underline{x}_{12}$ is a <u>solution</u> vector. However, $\underline{x}_{13}$ and $\underline{x}_{17}$ are <u>not</u> solutions. Now consider an enumeration scheme to determine all $\underline{x}_k$ such that $u_k \geq \omega$, where ω is a given constant. Sequentially select binary vectors $\underline{x}_k$ starting at the top of the list ($k = 2^n$), form u_k and work downward until $u_k < \omega$. This scheme will <u>not</u> guarantee that all $u_k \geq \omega$ have been found. It is not an acceptable alternative to the TPA.

(C) Associated with each <u>family</u> of solutions is a <u>range</u> of values

$$a \leq u(F_j) \leq b$$

instead of the single value u_k associated with an individual solution vector. Even though two families are disjoint, their ranges of $u(F_j)$ may be overlapping. For example, using Fig. 6-16, $8 \leq u(F_4) \leq 13$ and $7 \leq u(F_5) \leq 10$. It can be seen that if the range of a <u>canonical</u> family $F_k(\underline{x})$ is $u_k \leq u(F_j(\underline{x})) \leq u_\ell$, the range of the corresponding general form solution family $F_j(\underline{z})$ is given by $u_k - g \leq u(F_j(\underline{z})) \leq u_\ell - g$, where $g = -\sum_{(\gamma_j<0)} \gamma_j$. This is a convenient computational result.

6.74 The Use of Solution Families in a Document Retrieval System

We can identify each binary solution vector $\underline{z}_k$ with a particular combination of index terms. Each $\underline{z}_k$ may have m documents associated with it, m=0,1,2,···, and each of these m documents is predicted to be relevant.

A family of solutions $F_j(\underline{z})$ specifies a group of index term combinations which has relevant associated documents. The BRS consisting of the union of all the solution families will retrieve all documents from the file which are predicted relevant (have a utility $u \geq \tau$).

When the solution families $F_j(\underline{z})$ are considered with respect to a document retrieval system, several observations can be made about the usefulness of a BRS as derived from the LPBI.

(A) The BRS which consists of the union of $F_j(\underline{z})$ has the same exact form as the heuristically generated BRS which is the man-machine link in many existing systems. This provides a model with analytical end results which parallels the end results of a human being in current systems.

(B) The solution families $F_j(\underline{z})$ are mutually disjoint. This means that the BRS derived from the union of the $F_j(\underline{z})$ will never retrieve any document more than once. The BRS which is heuristically generated cannot be guaranteed to have this property.

(C) The cost of searching the file using a family of solutions $F_j(\underline{z})$ is much less than the cost would be if an equivalent search were run using each member $\underline{z}_k$ of the family separately.

(D) A disadvantage of searches made using solution families is that any documents retrieved by a solution family F_j can have a predicted utility spread over the range $a \leq u(F_j) \leq b$. and the predicted utility of a given retrieved document can be obtained exactly only with increased computation. The individual document utilities may be desired when a large number L of documents are cited as being relevant (predicted) by a BRS. The user may not have enough time to review all retrieved documents and may want only the subset of documents having the highest predicted utility. In this case the utility u_k can be determined for each document in the retrieved set by using the index term weights. The set of L documents can then be ranked and the N documents with the <u>highest</u> predicted utility presented to the user. In this case enumeration of document utilities is restricted to <u>only</u> the set of those predicted relevant, and this is usually a very small subset of the entire file.

6.75 Computer Implementation of the TPA

A computer program for solution of the LPBI using the TPA has been written in Fortran IV for the IBM 7094/7040 Direct Couple System.

Four subroutines control the solution of the LPBI and the output of data.

(A) The first subroutine forms the LPBI from the LUPF and converts all LPBI coefficients to integers. The LUPF as passed to this subroutine has real coefficients γ_j and no complemented variables ($\alpha_j = 1$ for all j). This subroutine converts all γ_j to integers by scaling and truncating. Accuracy of the conversion process is variable and is set by program parameters.

(B) The second subroutine transforms the integer LPBI parameters to canonical form, and finds all basic solutions to the canonical form. The basic solutions are written in groups of fixed size to an output device for temporary storage.

(C) The third subroutine produces canonical solution families from the basic solutions, and transforms the canonical solution families to get solution families to the general form LPBI. Basic solutions are read from the storage device to core in groups, are converted to general form solution families in core and then are again stored in groups on the output device. The range of $u(F_j)$ and SIZE $S(F_j) = \sum_{i=1}^{n} F_{ij}$ are also recorded with each solution family.

(D) The fourth subroutine will output solution families to a printer, or other device and which if desired will screen solution families on the basis of range or size and suppress printing of certain solution families if desired.

All four subroutines are under the exclusive control of a driver program. (No subroutine calls any other subroutine.) The resulting modular system is convenient to use and modify.

6.76 Computational Experience with the TPA

Experience with the TPA has been rather limited. Table 6-1 gives some performance data for 14 sample problems. The largest problem solved had only 14 variables.

By using these sample problems and by making some assumptions which seem reasonable based on the data of Table 6-1, rough estimates were obtained for larger problems. The assumptions are listed below.

(A) The number NV of nodes visited during solution of a LPBI increases exponentially with the number of problem variables n.

$$NV = A_o \, e^{\gamma_o n} \tag{6-24}$$

Parameters A_o and γ_o are experimentally determined constants.

(B) The number of basic solutions identified is proportional to the number of nodes visited.

$$BSOL = \alpha_2 NV \tag{6-25}$$

The parameter α_2 is a constant.

(C) The total number of solution points TS is, on the average, a fixed fraction α_1 of the total possible 2^n solution points.

$$TS = \alpha_1 2^n = \alpha_1 \, e^{0.693n} \tag{6-26}$$

TABLE 6-1

DATA ILLUSTRATING COMPUTATIONAL EXPERIENCE WITH THE TPA

Case number	n = number of variables	2^n	Nodes visited	Basic solutions	Degenerate basic solutions	Nondegenerate basic solutions	Total points in nondegenerate families	Total solution points
1	4	16	2	3	1	2	12	13
2	8	256	44	38	20	18	108	128
3	5	32	3	4	1	3	26	27
4	12	4096	206	318	100	218	3622	3722
5	3	8	2	2	1	1	2	3
6	8	256	17	18	8	10	200	208
7	4	16	3	2	1	1	2	3
8	10	1024	76	87	33	54	744	807
9	5	32	2	3	1	2	24	25
10	9	512	25	35	12	23	456	468
11	3	8	2	1	0	1	4	4
12	14	16384	632	733	317	416	14546	14863
13	4	16	4	3	2	1	2	4
14	12	4096	250	216	116	100	652	768

(D) A constant fraction α_3 of all basic solutions will be degenerate and $(1 - \alpha_3)$ will be non-degenerate.

$$\begin{aligned} BSOL &= DBSOL + NDBSOL \\ DBSOL &= \alpha_3 BSOL \\ NDBSOL &= (1 - \alpha_3) BSOL \end{aligned} \tag{6-27}$$

(E) The <u>average</u> number of solution points in a non-degenerate solution family is an increasing function of n, FS(n). The analytical form of this function can be derived from assumptions (A) - (D) above as follows.

For the total number of solutions we can write two equivalent expressions;

$$TS = \alpha_1 \; e^{0.693n} \tag{6-26}$$

and

$$\begin{aligned} TS &= DBSOL + (NDBSOL)FS(n) \\ &= \alpha_3\alpha_2 A_o e^{\gamma_o n} + \left[(1 - \alpha_3)\alpha_2 A_o \; e^{\gamma_o n}\right] FS(n) \end{aligned} \tag{6-28}$$

Equating (6-26) to (6-28) and solving for FS(n) gives

$$FS(n) = \frac{\alpha_1 \; e^{0.693n} - \alpha_2\alpha_3 A_o \; e^{\gamma_o n}}{\alpha_2(1 - \alpha_3)A_o \; e^{\gamma_o n}} \tag{6-29}$$

TABLE 6-2

SMOOTHED AND EXTRAPOLATED ESTIMATES OF TPA PERFORMANCE

A. Estimated Solution Time as a Function of Problem Size

Number of variables	Expected node visits	Total node visiting time (sec) at node visiting rates (R) shown below (nodes/sec)		
n	NV	R = 500	R = 1000	R = 2000
10	85	0.17	0.0850	0.0425
15	121	2.42	1.21	.605
20	17,800	35.60	17.3	8.65
25	250,000	500.	250.	125.0
30	3,627,000	7260. (121 min)	3630. (60.5 min)	1815. (30.25 min)

B. Number and Type of Solutions as a Function of Problem Size

Number of variables	Basic solutions	Nondegenerate basic solutions	Degenerate basic solutions	Average sols/fam	Total solutions in families	Total number solutions
n	BSOL	NDBSOL	DBSOL	FS(n)	(NDBSOL)FS(n)	TS
10	98	59	39	10.3	6.08×10^2	6.47×10^2
15	1,400	840	560	23.4	1.97×10^4	2.03×10^4
20	20,000	12,350	8,250	52.4	6.47×10^5	6.55×10^5
25	289,000	173,500	115,500	117.6	2.03×10^7	2.04×10^7
30	4,200,000	2,520,000	1,700,000	260.3	6.56×10^8	6.58×10^8

From the data of Table 6-1, estimates of the parameters are:[1]

$$\begin{aligned} \alpha_1 &= 0.622 \\ \alpha_2 &= 1.155 \\ \alpha_3 &= 0.400 \\ A_o &= 0.403 \\ \gamma_o &= 0.5345 \end{aligned} \tag{6-30}$$

and it follows that (6-29) then becomes:

$$FS(n) = 2.23\ e^{0.1585n} - 0.67\ . \tag{6-31}$$

The results of applying the above assumptions (6-24) to (6-31) for selected values of n are shown in Table 6-2B. If one computer word is used to store each basic solution, it appears that the storage problem for the 25 variable problem is excessive, with 289,000 basic solutions expected. The 20 variable problem appears more reasonable, with 20,000 basic solutions expected.

Table 6-2A shows the expected processing time based on three different average node visiting rates. The current node visiting rate is about 500 nodes/second. With some very trivial program modifications, this can be extended to 1000 nodes/second or above. The 25 variable problem at 1000 nodes/second will require an estimated 250 seconds for solution. This is considered excessive, and the 20

---

[1] The reader is cautioned that the variances of the parameter estimates are quite large. Smoothed and extrapolated data based on these parameters is intended for rough estimates only. Data is also peculiar to the application here, where index term weights are derived using approximation theory.

variable problem again appears more reasonable with a 17.3 second total.

Times given are for the TPA which finds basic solutions to the canonical form. Subroutines which transform parameters to canonical form and which transform basic solutions to general solution families require much less time than the TPA. Their contribution to total processing time is ignored here.

In conclusion, the TPA appears adequate for solving LPBI's with up to 20 variables. For the 20 variable problem, the expected processing time is 17.3 seconds (at a node visiting rate of 1000 nodes/second). For the same problem, expected storage space is 20,600 words, assuming one basic solution per word. Both solution time and storage requirements appear reasonable for applications related to document retrieval systems.

7.0 EXPERIMENT DESIGN AND PRESENTATION OF DATA

This chapter discusses the experiment design configuration selected for test purposes and presents the raw response data. Test objectives and the various measures of search effectiveness are also discussed. Analysis of the experimental data is deferred to chapter 8.0.

7.1 Test Objectives

The test program had three objectives.

(A) First, to determine whether significant differences in search effectiveness exist between searches performed using machine-generated BRS's and searches using BRS's generated heuristically by humans.

(B) Secondly, to help determine the causes of these differences, if they exist.

(C) Finally, to provide an overview of the whole process and suggest areas for further research.

Before presenting test details, it is convenient to discuss figures of merit used to evaluate the effectiveness of document retrieval systems.

7.2 Measuring Search Effectiveness

Three measures of effectiveness are used here to evaluate test results. All are based on entries in the following 2 x 2 contingency table.

$$
\begin{array}{l|c|c|c}
 & \text{Retrieved} & \text{Not Retrieved} & \\
\hline
\text{Relevant} & n_{11} & n_{12} & n_{1\cdot} \\
\text{Not Relevant} & n_{21} & n_{22} & n_{2\cdot} \\
\hline
 & n_{\cdot 1} & n_{\cdot 2} & N
\end{array}
\qquad (7\text{-}1)
$$

For each search, a contingency table identical to (7-1) can be constructed. This assumes that all relevant documents are known, whether retrieved or not.

7.21 Recall and Precision

Two standard measures of search effectiveness based on the contingency table are <u>recall</u> and <u>precision</u>. These measures have been proposed and used by several authors (94,95).

The definitions are:

$$\text{Recall} = \frac{n_{11}}{n_{1\cdot}} = \frac{\text{relevant retrieved}}{\text{total relevant}} \, . \qquad (7\text{-}2)$$

$$\text{Precision} = \frac{n_{11}}{n_{\cdot 1}} = \frac{\text{relevant retrieved}}{\text{total retrieved}} \, . \qquad (7\text{-}3)$$

Roughly, recall is a measure of how well the system retrieves all the relevant material, while precision is a measure of the economy of the retrieval process. Variations of the above definitions of recall and precision are occasionally used. See, for example, Salton[96].

7.22 The Information Statistic as a Measure of Search Effectiveness

A disadvantage of recall and precision is that a pair of numbers are involved instead of a single figure of merit. An alternate measure based on the 2 x 2 contingency table has been proposed and used by A. R. Meetham,[97] which gives a single figure of merit for the search effectiveness.

It is identical to the information measure $\hat{R}$ described in chapter 4;

$$\hat{R} = H(X) - H(X/Y) = \frac{1}{N}\sum_{i}\sum_{j} n_{ij} \log \left(\frac{N\, n_{ij}}{(n_{i.})(n_{.j})}\right). \qquad (4\text{-}8)$$

This computational formula was derived in section 4.25.

Recall that $\hat{R}$ is the gain in information (reduction in entropy) which occurs (on the average) each time new information $p(Y)$ is used to convert a prior distribution $p(X)$ to a posterior distribution $p(X/y)$. The prior distribution $p(X)$ is an initial assignment of probabilities to states of nature and the posterior distribution $p(X/y)$ is the revised probability distribution after observing auxiliary data, or the results of an experiment y. (See chapter 4.0.)

The information measure $\hat{R}$ is used in chapter 4 to select the (most discriminating) index terms for inclusion in the decision function. $\hat{R}$ is used here to evaluate document retrieval system effectiveness. This allows a new view of the retrieval process as a prior to posterior probability distribution adjustment. The prior distribution is the probability of a document in the file being relevant, given that it is drawn at random, and with no knowledge of index terms etc., which are associated with the document. The posterior distribution is the probability that a document which is selected by the retrieval system is relevant. (This selection is based on the index terms.)

The retrieval system can be viewed as an automatic processor which performs an auxiliary experiment on the index terms associated with a document, and then by using a built-in decision rule on these experimental results, offers a suggestion to the user as to whether the document is relevant or not. After seeing the document the <u>user</u> makes a final decision about its relevance. The degree of agreement which exists between the judgements made by the retrieval system and the user is the measure $\hat{R}$ of how well the system operates.

A perfect retrieval system would make decisions (suggestions) about document relevance which would always agree with the user judgement. The system suggestion would then <u>remove</u> <u>all</u> uncertainty (for the user) about document relevance. In this case $\hat{R} = H(X)$. Any real system of course will not be perfect. As a consequence we will have $0 \leq \hat{R} \leq H(X)$.

Define:

$$\alpha = 100[\hat{R}/H(X)]. \tag{7-4}$$

Then we have:

$$0 \leq \alpha \leq 100.$$

The variable α is the normalized information statistic (NIS) and is interpreted as the percent effectiveness of the retrieval system. It can be thought of as the average percent reduction in uncertainty about document relevance, if the system suggestion regarding document relevance is followed. The measure (7-4) will be used in the experiment described here to evaluate the retrieval system, in addition to recall (7-2) and precision (7-3).

The relation of the NIS to recall and precision is shown in Fig. 7-1, for a file similar to the one used for test purposes. It can be seen that recall and precision are both strictly increasing functions of the NIS. Thus, increasing the NIS will never degrade either recall or precision.

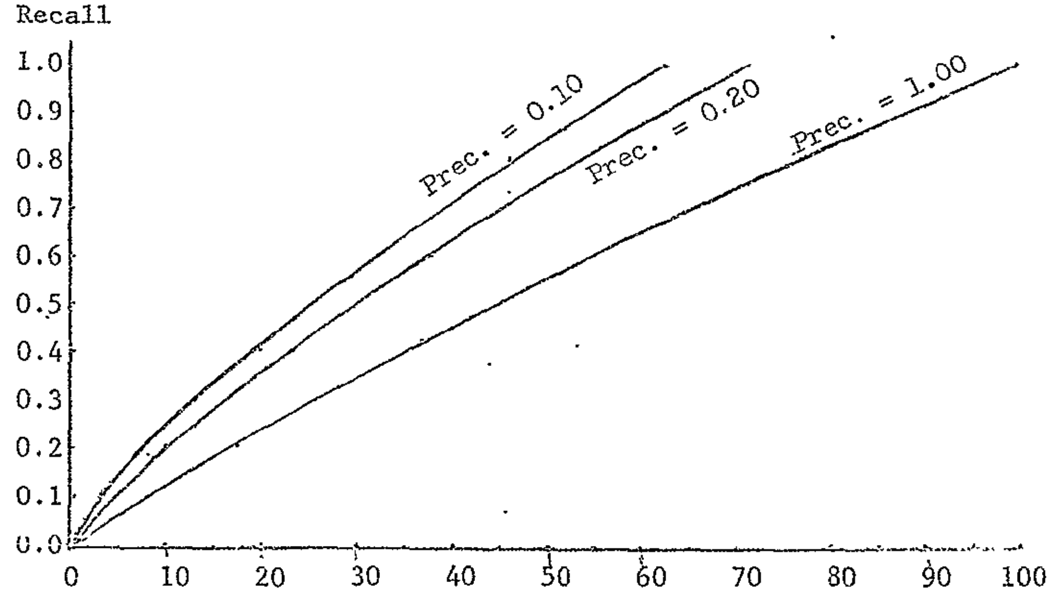
7.23 Other Applications of the Information Statistic

The NIS as described here was used by Shirey[98] to evaluate the efficiency of document abstracts and first - last paragraph combinations at predicting document relevance. After reading these relevance cue indicators, the users were asked to make a judgment about the relevance of the full document. After this first judgment was

FIGURE 7-1

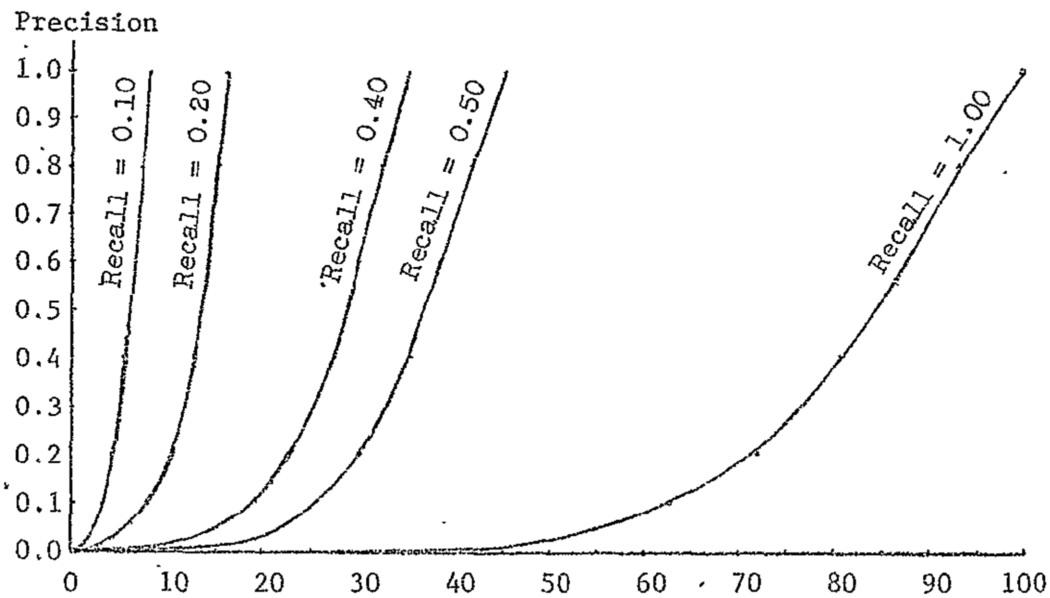
RECALL AND PRECISION VS. THE NORMALIZED INFORMATION STATISTIC (NIS) FOR $N = 5100$ AND $n_{11} = 2$

A. Recall vs. NIS



NIS = $100\,\alpha = 100[R/H(x)]$

B. Precision vs. NIS



NIS = $100\,\alpha = 100[R/\hat{H}(X)]$

obtained, the users were shown the full document and asked for a second final opinion of relevance. The preliminary and final results were analyzed and $\hat{R}/H(X)$ was computed. In this application the use of relevance cue indicators constitutes an <u>experiment</u> performed to provide more information about document relevance.

R. H. Shumway[99] has also noted the potential use of the information statistic $\hat{R}$ as an overall measure of retrieval system effectiveness. In addition, he re-analyzes the Shirey data assuming a <u>three-way</u> table relation. He demonstrates that the two-way table used by Shirey for his analysis is really a special case of multi-way contingency tables. These can be analyzed using an information measure which is partitioned in a manner similar to the sum of squares in the analysis of variance. The general method is treated by Kullback[100].

7.24 Summary

The information statistic $\hat{R}$ described above was developed in chapter 4 for selection of index terms (a form of feature extraction). It is used again here in its normalized form (7-4) as a figure of merit for evaluating retrieval systems.

It has been both used and proposed by others for extraction of pattern features (see section 1.4), evaluation of search effectiveness, evaluation of relevance cue indicators, and general contingency table analysis.

7.3 Experiment Design

7.31 General

A 2^3 factorial experiment was designed to determine whether retrieval system effectiveness is influenced by:

(a) methods (BRS's generated by machine vs. BRS's generated by people);

(b) number of index terms used in the model (a high level of about 15 terms and a low level of about 5 terms); and

(c) number of documents in the training set (50 documents at a high level and 25 documents at a low level).

The 2^3 factorial configuration was replicated four times, with each replication (of $2^3 = 8$ points) being a separate query to the system. This allowed variability existing between questions to be accounted for.

One month of the NASA file (a total file size of 4881 documents) was searched using the different BRS's. All the documents relevant to the four queries were identified before the searches were performed. The figures of merit for each search were then computed from the 2 x 2 contingency tables (7-1) constructed after completion of the searches.

7.32 Selection and Preparation of Test Questions

The four questions used as replicates were selected at random from a group of actual queries in an information system. Each question selected had an existing associated group of abstracts rated relevant or non-relevant by a user. There were enough existing abstracts to construct a 50 document training set.

Before the training set was finalized, the month to be searched (March, 1969) was chosen at random, and all abstracts in the training set for this month were removed. The 50 abstracts remaining in the training set were from within six months before and after the search month of March, 1969.

By using the training set abstracts, a detailed question description was written. Nine meaningful and identifiable subcategories for each question were devised, and each subcategory was assigned a utility from 1 to 9. Each of the 50 abstracts was then placed in one of the nine subcategories, and a utility threshold τ was introduced which designated which of the subcategories were relevant and which were not. With the questions well defined by the training sets and the written descriptions, a complete manual search was performed over the March, 1969 portion of the file and all relevant documents for each query were identified.

A 25-document training set was created for each question by selecting 25 documents from each 50-document training set. (The 50 documents were ordered sequentially by their file numbers, and then every other number was chosen. Since file numbers are unrelated to utilities, this selection method is believed unbiased.) This gave eight training sets, one of 50 and one of 25 documents for each of the four test questions. Preparations for testing were completed by assembling a 'package' for each of the eight training sets. This package consisted of:

(a) a sequential listing of all document numbers, the utility assigned to each, and the set of associated index terms;

(b) the utility threshold τ defining relevance;

(c) full abstracts of each training set document, grouped by utility sub-category, with each group also marked as being relevant or non-relevant;

(d) one-sentence abstracts of each training set document, grouped and marked as in (c) above.

The 32 experimental BRS's were next derived using the above training sets. For each of the eight training sets two BRS's were constructed; one using five index terms and the other fifteen. This was repeated for two methods of BRS construction (machine and analyst) to give a total of 32 combinations.

The machine generated BRS's (16 of them) were constructed using the methods described in previous chapters. First, best single index terms were selected. Next, the LUPF was fit to the assigned document utilities. Finally, using the utility threshold, the LPBI was formed and all solution families were found. Only items (a) and (b) in the training set packages were utilized by the machine system.

Another 16 BRS's were constructed <u>heuristically</u> by four experienced information analysts. Each analyst was assigned one particular combination of training set size (25 or 50) and number of index terms (5 or 15) for each question. There are four such combinations per question; one per analyst. Each analyst was assigned only <u>once</u> to each of the four combinations.

The analyst was then requested to construct a BRS for this particular combination. The effect of different analysts is considered to be an integral part of the subjective method (method 1). The analysts utilized items (a), (b), (c) and (d) of the training set packages. They were not, however, given the question description. They were required to infer the question meaning by reading the abstracts for relevant and non-relevant documents and by noting the utilities assigned to each abstract. Each analyst was given a maximum of one hour to write the BRS assigned to him.

Finally, the file was searched using each of the 32 BRS's. Searches, using the BRS's generated by the information analysts were made with an existing computer program. The machine-generated BRS's were not used directly as search instructions. Instead, the equivalent sets of index term weights were used.

7.33 Classification of Variables in the Problem

It is convenient to place the problem variables into three groups.

7.331 Independent Variables Controlled as Part of the Problem. This includes Methods (M), where M_1 is the subjective method using human analysts and M_2 is the machine method. This factor is fixed and qualitative. Analysts appear implicitly as part of this factor.

Also controlled were the number of index terms (T) appearing in the training set. T_1 refers to the lower level (about 5 terms) and T_2 refers to the higher level of about 15 terms. This factor is fixed and qualitative because the number of terms varied slightly, but was identifiable at either a high or low level.

Documents in the training set (D) were run at two levels. A lower level D_1 of 25 documents and an upper level D_2 of 50 documents was used. Factor D is fixed and quantitative.

Questions (Q) were run as the replication (or block) variable to lower the error variance. The entire experiment is conveniently classified as a 2^3 factorial run in a randomized block design.[101] Four questions (replications or blocks) were used. Factor Q is random and qualitative.

7.332 Variables Held Constant as Boundary Conditions on the Problem. This includes the fraction of the training set which is relevant (about 50%), the time allowed each analyst to construct the BRS, and the method of query presentation to the analyst. Other variables held constant are the extent of file searched (one month, or 4881 documents) and the particular time period of the file (March, 1969).

7.333 Uncontrolled Factors Contributing to the Error Variance. In this group are the system indexing, compatibility of the question to the system, and consistency of the question itself. Also, the variation between analysts within method 1 contributes to error variance.

7.34 Factors and Variables Not Considered in the Experiment

The following important items were not considered in this experimental program.

(A) The effect of adaptive refinement of the BRS through additions and/or deletions from the training set, followed by repeated searches, was not investigated. BRS's refined over several searches by supplementing the training set would be expected to produce better results than the BRS's used here.

(B) Experienced information analysts were used to construct BRS's for test purposes instead of casual system users. The effect of user experience was not investigated, but casual users would not be expected to construct BRS's which would be as effective as those of more experienced users.

(C) Most of the L_1 problems solved for index term weights (M_2) exhibited alternate optimal solutions (see section 5.5). The retrieval efficiency of these alternate optimal solutions was not investigated. The initial optimal solution was always used for retrieval purposes.

7.35 The Model Equation and Expected Mean Squares Table

The model equation for the factorial experiment is given by:

$$y_{ijk\ell} = \mu + T_i + D_j + TD_{ij} + M_k + TM_{ik} + DM_{jk} + TDM_{ijk} + Q_\ell + \varepsilon_{ijk\ell} \tag{7-5}$$

where $i = 1,2$ and $\begin{cases} i=1 \Longrightarrow 5 \text{ index terms} \\ i=2 \Longrightarrow 15 \text{ index terms} \end{cases}$;

$j = 1,2$ and $\begin{cases} j=1 \Longrightarrow 25 \text{ documents in training set} \\ j=2 \Longrightarrow 50 \text{ documents in training set} \end{cases}$;

$$k = 1,2 \quad \text{and} \quad \begin{cases} k=1 \Longrightarrow \text{method 1 (analysts produce BRS)} \\ k=2 \Longrightarrow \text{method 2 (machine produced BRS)} \end{cases}; \text{ and}$$

$\ell = 1,2,3,4$ for 4 questions each functioning as a replication.

All factors except Q are fixed. Q is a random factor. The expected mean square table is shown below[102].

Factor	Fixed or random	Degrees of freedom	Expected mean square
T(index terms)	F	1	$\sigma_e^2 + 16\sigma_T^2$
D(documents)	F	1	$\sigma_e^2 + 16\sigma_D^2$
TD	F	1	$\sigma_e^2 + 8\sigma_{TD}^2$
M(methods)	F	1	$\sigma_e^2 + 16\sigma_M^2$
TM	F	1	$\sigma_e^2 + 8\sigma_{TM}^2$
DM	F	1	$\sigma_e^2 + 8\sigma_{DM}^2$
TDM	F	1	$\sigma_e^2 + 4\sigma_{TDM}^2$
Q(questions)	R	3	$\sigma_e^2 + 8\sigma_Q^2$
error	R	21	σ_e^2

(7-6)

Note that an exact F test exists for each of the effects in terms of the error mean square.

7.36 Choice of Sample Size

The sample size was determined by choosing acceptable risk levels associated with the test for a difference between treatment means for main effect M (methods). This test between means is summarized by the following hypotheses:

$$H_o: \ |NIS(M_1) - NIS(M_2)| = 0 \tag{7-7}$$

$$H_1: \ |NIS(M_1) - NIS(M_2)| > 0. \tag{7-8}$$

Here $NIS(M_1)$ and $NIS(M_2)$ are the true mean values of the normalized information statistic for method 1 (subjective) and method 2 (machine).

Estimates of the mean and variance of the NIS for a one month search of the NASA file were first determined subjectively. These estimates were 28.3 percent for the average NIS and $.177 = \sigma_S^2$ for the NIS variance.

The test statistic for a difference δ between NIS treatment means is given as:[103]

$$t = \left[\frac{[\overline{NIS}(M_1) - \overline{NIS}(M_2)] - \delta}{\sqrt{S_e^2\left(\frac{1}{r_1} + \frac{1}{r_2}\right)}}\right] \tag{7-9}$$

which is distributed as Student's t with ν degrees of freedom where:

(a) $\overline{NIS}(M_1)$, $\overline{NIS}(M_2)$ are the treatment means (<u>average</u> NIS responses for methods M_1 and M_2);

(b) r_1, r_2 are the number of replications in each treatment mean;

(c) S_e^2 is the error variance (of the NIS response) as estimated from the experiment;

(d) δ is the true difference between the treatment means (difference between NIS responses for methods M_1 and M_2); and

(e) ν is the number of degrees of freedom in S_e^2.

The null hypothesis (7-7) now becomes H_o: $\delta = 0$.

After some deliberation, it was decided that a true difference $\delta = 10\%$ between NIS response to the two different treatments would be meaningful to retrieval system operation and should be detected by the experiment. Also, the type I error (alpha) was fixed at 0.10. Because the cost in both time and effort of experimentation is great, a compromise was reached for four replications of the 2^3 factorial, or 32 data points (searches). This gave $r_1 = r_2 = 16$; $\nu = 21$; and $\sigma_e = 13.3$, which is the previous subjective estimate of the NIS standard error.

An operating characteristic curve constructed for the t-test (7-9) using this data is summarized below, where the type II error (beta) or not detecting a true difference δ is given as a function of δ.

True difference $\delta = (R/H)100$	Type II error (beta)
0	0.90
2	.81
5	.62
10	.23
15	.05
20	.01

(7-10)

In summary, for the chosen configuration, it can be seen that if the true $\delta = 10$, the probability of not detecting this difference is 0.23 (the beta error). Alternately, there is a probability of 0.10 (the alpha error) of falsely detecting a significant difference, given that there is none.

7.37 A Sub-Experiment to Determine the Effect of Analysts within Method 1

Analysts are considered to be an integral part of method 1 for the 2^3 factorial experiment. However, when method 1 is considered alone, it is meaningful to isolate the effects of the analysts.

To consider this effect, it was necessary to control the arrangement of analysts, questions and treatments within method 1. This was done with a latin square configuration[104]. The model equation is:

$$y_{ijk} = \mu + A_i + Q_j + T_k + \varepsilon_{ijk} \qquad (7\text{-}11)$$

where A_i: i=1,2,3,4 are analysts;

Q_j: j=1,2,3,4 are questions;

and T_k: k=1,2,3,4 are treatment combinations.

Figure 7.2A shows the particular latin square configuration chosen. One combination shown in this figure is query 1 (Q_1) with analyst 3 (A_3) using treatment 4 (T_4), which consists of a training set of 50 documents and a BRS with 15 index terms.

7.4 Presentation of the Experimental Data

7.41 Factorial Experiment Response Data

Table 7-1 shows the response data from the 32 experimental searches. Each response is given in terms of contingency table entries. This is followed by the NIS, the recall and the precision of the search, all of which are computed from the contingency table.

For example, consider data point 8 of table 7-1 (read across line 8). This point corresponds to a search with a BRS formed using nominally 15 index terms ($T_2 = 15$); from a training set with 50 documents ($D_2 = 50$); using method 2 (M_2 for a machine BRS); and searching query 1 (Q_1). The corresponding 2 x 2 contingency table is given by:

	Relevant	Not relevant		
Relevant	3	9	12	
Not relevant	29	4840	4869	(7-12)
	32	4849	4881	

This table corresponds to (7-1). The NIS is computed from (7-12) using the same methods presented in the example of Fig. 4-2, and described in section 4.55. For search 8, the NIS is 10.15 (percent). The search recall and precision are also computed from (7-12) by using (7-2) and (7-3). These are given as 0.333 and 0.094, respectively.

TABLE 7-1. - RESPONSE DATA FROM FACTORIAL EXPERIMENT

Data point	Treatment combination				Contingency table				Response		
	Q	M	D	T	n_{11}	n_{12}	n_{21}	n_{22}	NIS	Recall	Precision
1	1	1	25	5	3	9	19	4850	11.54	0.250	0.136
2				15	4	8	27	4842	15.35	.333	.129
3			50	5	10	2	21	4848	55.90	.833	.323
4				15	6	6	34	4835	25.00	.500	.150
5		2	25	5	8	4	27	4842	39.10	0.667	0.229
6				15	5	7	26	4843	20.94	.420	.161
7			50	5	4	8	23	4846	16.05	.333	.148
8				15	3	9	29	4840	10.15	.333	.094
9	2	1	25	5	2	2	3	4874	35.35	0.667	0.500
10				15	2	2	3	4874	35.35	.667	.500
11			50	5	2	2	9	4868	29.65	.667	.500
12				15	2	2	79	4798	16.90	.667	.053
13		2	25	5	0	4	74	4803	0.19	0.000	0.000
14				15	0	4	116	4761	.30	.000	.000
15			50	5	0	4	60	4817	.15	.000	.000
16				15	3	1	114	4763	27.77	1.000	.036
17	3	1	25	5	4	8	90	4779	9.98	0.333	0.042
18				15	1	11	3	4866	4.53	.083	.250
19			50	5	8	4	115	4754	26.36	.667	.065
20				15	1	11	27	4842	2.14	.083	.036
21		2	25	5	1	11	25	4844	2.22	0.083	0.038
22				15	4	8	136	4733	8.15	.333	.028
23			50	5	8	4	227	4642	20.16	.667	.034
24				15	6	6	247	4622	11.68	.500	.024
25	4	1	25	5	0	5	1	4875	0.00	0.000	0.000
26				15	2	3	2	4874	29.08	.400	.500
27			50	5	0	5	0	4876	0.00	.000	.000
28				15	3	2	1	4875	49.65	.600	.750
29		2	25	5	0	5	68	4808	0.18	0.000	0.000
30				15	0	5	127	4749	.34	.000	.000
31			50	5	0	5	69	4807	.18	.000	.000
32				15	1	4	73	4803	4.45	.200	.014
					Mean responses:				15.90	0.353	0.148

7.42 Latin Square Experiment Response Data

Table 7-2B gives the response data (NIS only) for this experiment. For example, the BRS submitted by analyst 2 for question 3 resulted in a search having an NIS response of 26.36. (This is search 19 of table 7-1.)

TABLE 7-2. - RESPONSE DATA FOR LATIN SQUARE DESIGN WITHIN METHOD 1

A. Latin Square Layout

	Q_1	Q_2	Q_3	Q_4
A_1	T_1	T_3	T_4	T_2
A_2	T_2	T_4	T_3	T_1
A_3	T_4	T_1	T_2	T_5
A_4	T_3	T_2	T_1	T_4

Treatment definitions:

T_1 = 25/5
T_2 = 25/15
T_3 = 50/5
T_4 = 50/15

B. Latin Square Response Data (NIS)

	Q_1	Q_2	Q_3	Q_4
A_1	11.54	29.65	2.14	29.08
A_2	15.35	16.90	26.36	0.003
A_3	25.00	35.35	4.53	0.000
A_4	55.90	35.35	9.98	49.650

7.43 Predicted Document Utilities vs. Known Document Utilities

Half (16) of the experimental searches were performed using a machine derived BRS (M_2). Recall that either the index term weights or an equivalent BRS can be used to search the file. For the 16 M_2 data points, the file was searched using the term weights. This was done as a matter of practical convenience. (The equivalent BRS's were also derived and will be discussed in section 7.45.)

When searching with term weights, a predicted utility $\hat{u}$ is computed for each document in the file. Because the utility threshold τ varies from question to question, it is convenient to compare predicted utilities by using $(\hat{u} - \tau)$ instead of $\hat{u}$. Here $(\hat{u} - \tau) \geq 0$ if the document is predicted to be relevant and $(\hat{u} - \tau) < 0$ otherwise.

When weighted index term searches are performed on a file of documents, any given document from the file ends up in one of three categories.

(A) No index terms with assigned weights match index terms in the given document.

(B) One or more of the index terms associated with the given document matches index terms in the search strategy, and $(\hat{u} - \tau) \geq 0$. (Relevance is predicted.)

(C) One or more of the index terms associated with the given document matches index terms in the search strategy, and $(\hat{u} - \tau) < 0$.

For the 16 weighted term searches, an average of 5.72 percent of all documents fell into categories (B) or (C) above; 3.82 percent had $(\hat{u} - \tau) < 0$ and 1.90 percent had $(\hat{u} - \tau) \geq 0$. For the file of 4881 documents, this gave an average per-search yield of 93 documents with $(\hat{u} - \tau) \geq 0$ (category B) and 186 documents with $(\hat{u} - \tau) < 0$ (category C).

Table 7-3A shows the relative frequencies $P(\hat{u} - \tau)$ of the predicted utilities for all M_2 searches in categories (B) or (C) above. Since coefficients of the LUPF are integral multiples of 1/2, so are the values of $(\hat{u} - \tau)$. (See section 5.7.) For example, 16.45 percent of all documents in categories (B) or (C) had predicted utilities of -3.0 <u>or</u> -2.5. The distribution of $P(\hat{u} - \tau)$ tends to be bimodal, having separate modes for the documents with $(\hat{u} - \tau) \geq 0$, and for those with $(\hat{u} - \tau) < 0$.

Relevant documents in the file had been identified and assigned utilities before the searches were run. It is possible to compare the <u>pre-assigned</u> values of $(u - \tau)$ for these relevant documents with the values of $(\hat{u} - \tau)$ <u>predicted</u> by the system.

Tables 7-3B and 7-3C compare the <u>predicted</u> $(\hat{u} - \tau)$ with the <u>assigned</u> $(u - \tau)$ for the relevant documents only. Table 7-3B gives a coarse cross-classification showing $(\hat{u} - \tau)$ grouped into categories (A), (B) or (C) above. For example, 13 relevant documents with an assigned $(u - \tau) = 1$ were placed by the 16 M_2 searches into category (B). There were a total of 132 relevant documents associated with the group of 16 M_2 searches.

Table 7-3C gives a more detailed breakdown of cross classification information contained in table 7-3B. For example, three relevant

TABLE 7-3. - COMPARISON OF PREDICTED DOCUMENT UTILITY WITH ACTUAL DOCUMENT UTILITY

A. Relative Frequencies of Observed Values of $(\hat{u} - \tau)$

$\hat{u} - \tau$	$P(\hat{u} - \tau)$	$\hat{u} - \tau$	$P(\hat{u} - \tau)$
-8.0	0.0007	0.0	0.0459
-7.0	.0013	1.0,1.5	.0985
-6.0	.0018	2.0	.1410
-5.0	.1014	3.0	.0378
-4.0	.1072	4.0	.0054
-3.0,-2.5	.1645	5.0	.0009
-2.0,-1.5	.1386	6.0	.0020
-1.0,-0.5	.1518	7.0	.0007
		8.0	.0004

B. Comparison of Preassigned Document Utilities $(\hat{u} - \tau)$ With Those Predicted by the Linear Model $(\hat{u} - \tau)$ for Relevant Documents

Predicted Utilities (Coarse)

		No match (A)	$(\hat{u} - \tau) \geq 0$ (B)	$(\hat{u} - \tau) < 0$ (C)	
True Utilities $(u - \tau)$	0	23	19	6	48
	1	22	13	9	44
	2	15	7	6	28
	3	4	5	3	12
		64	44	24	132

C. Detailed Breakdown of Table 7-3B Above

Predicted Utilities

$(\hat{u} - \tau)$

		-4	-3	-2	-1	0	1	2	3	4	5	6	(A)	
True Utilities $(u - \tau)$	0			2	4	13	2	1	0	3			23	48
	1			4	5	1	3	3	5			1	22	44
	2	1		3	2	2	1	2	2				15	28
	3			2	1	1	1	2	1				4	12
		1	0	11	12	17	7	8	8	3	0	1	64	132

documents had a predicted utility $(\hat{u} - \tau) = 4$. All four of these documents had an assigned or true $(u - \tau) = 0$.

7.44 Values of $\hat{R}$ for Index Terms in the Training Sets

Table 7-4 shows the distribution of $\hat{R} = H(X) - H(X/Y)$ for the index terms which appeared with documents in the eight different training sets used for the experiment. An average of 148 different terms were found with each 25 document training set and an average of 250 terms were found with each 50-document training set. To illustrate the use of table 7-4, there are two index terms with $0.15 \leq \hat{R} \leq 0.19999$ in the 25 document training set (D = 25) associated with query 1 (Q_1).

TABLE 7-4. - DISTRIBUTION OF $\hat{R} = H(X) - H(X/Y)$ FOR INDEX TERMS APPEARING IN THE TRAINING SETS

$\hat{R}$(bits)	D = 25					D = 50			
	Q_1	Q_2	Q_3	Q_4		Q_1	Q_2	Q_3	Q_4
0.00 - 0.04999	110	91	121	141		195	210	280	254
.05 .09999	7	28	36	23		11	15	10	13
.10 .14999	8	9	6	2		2		6	1
.15 .19999	2		3	2	95 %			1	1
.20 .24999					conf.				
.25 .29999	1							1	
Total terms	128	128	166	168		208	225	298	269

7.45 BRS Descriptions

The BRS is a union of index term solution families. It is convenient to describe a BRS by using some particular attribute of the solution families from which it is formed. One useful attribute of an individual solution family is the number of index terms which must be simultaneously present in a document to cause the document to match the family and hence be retrieved. This attribute will be called the SIZE (S) of the family and will be used to compare machine-generated BRS's with those heuristically generated by users.

Let the SIZE S of a solution family be the number of fixed variables F_{kj} which equal unity in the family F_k. That is

$$S = \sum_{j=1}^{n} F_{kj} \quad \text{for} \quad F_{kj} \neq (-) \tag{7-13}$$

The following simple example illustrates this definition.

Family	$(T_1T_2T_3T_4)$	SIZE(S)
F_1	(1,0,-,1)	2
F_2	(0,1,-,-)	1
F_3	(1,1,1,-)	3

(7-14)

Families with $S = 1$ are those which specify the presence of only one matching index term in order to retrieve the document. Families with $S = 2$ require a specified pair of index terms to be present. Note that variables in the family which are fixed at zero require the absence of the corresponding index term in order that the document will be retrieved.

Table 7-5 shows the distribution of solution families having a size S within a BRS for 30 of the 32 BRS's used in the experiment.

(BRS data was lost for data points numbered 29 and 30.) For example, consider data point 14 (question 2, M_2 (machine), a 25 document training set, with a nominal 15 index terms used for the BRS). There were 12 solution families in the associated BRS. Four of these families had $S = 1$, six had $S = 2$ and two had $S = 3$.

TABLE 7-5. - DISTRIBUTION OF SOLUTION

FAMILIES HAVING SIZE S

Data point	Treatment combination					Solution family size									
	Q	M	D	T	L*	1	2	3	4	5	6	7	8	9	10
1	1	1	25	5	5	0	6								
2				15	14	0	45								
3			50	5	6	6	0								
4				15	13	0	64								
5		2	25	5	4	3									
6				15	8	3	8	11	10	4	2				
7			50	5	5	2	2								
8				15	12	1	36	64	105	77	35				
9	2	1	25	5	5	0	6								
10				15	16	0	22								
11			50	5	5	0	6								
12				15	15	0	56								
13		2	25	5	2	2									
14				15	7	4	6	2							
15			50	5	3	2									
16				15	10	5	16	26	17	13	8	2			
17	3	1	25	5	6	4	1								
18				15	15	0	56								
19			50	5	5	5	0								
20				15	14	0	54								
21		2	25	5	5	2	0	1							
22				15	9	3	10	12	6	4					
23			50	5	3	1									
24				15	14	4	29	70	108	154	157	123	60	26	2
25	4	1	25	5	5	0	6								
26				15	15	0	50								
27			50	5	5	0	6								
28				15	18	0	42								
29		2	25	5	4	–	––	––							
30				15	11	–	––	––							
31			50	5	4	1	1	––							
32				15	12	3	16	27	36	65	26	30	12		
Methods totals		M_1	435			15	420	--	--	--	--	--	--	--	--
		M_2	1515			36	124	213	282	317	228	155	132	26	2

*L is the actual number of index terms in the BRS.
T = 5 or 15 is the nominal number

8.0 EXPERIMENTAL DATA ANALYSIS

8.1 Analysis of Variance for the Factorial Experiment

The response data for each of the 32 experimental searches appears in Table 7-1. Three different measures of search effectiveness are considered (NIS, precision and recall). The experimental data of Table 7-1 is analyzed separately for each measure of effectiveness. Three corresponding analysis of variance (ANOVA) tables are shown in Table 8-1. These will be discussed below. Only effects which are significant at an alpha level of at least 0.10 (confidence level of 90%) will be discussed.

8.11 Dependence of the NIS on Methods

The ANOVA table for this measure of effectiveness is shown in Table 8-1A. The only factor significantly affecting the NIS is that of search methods (M). Heuristic BRS's (M_1) gave better results than did machine BRS's (M_2). The experiment treatment means are:

$$\overline{NIS}(M_1) = 21.67$$

$$\overline{NIS}(M_2) = 10.13$$

$$\overline{\Delta} = \overline{NIS}(M_1) - \overline{NIS}(M_2) = 11.54 .$$

TABLE 8-1. - ANALYSIS OF VARIANCE TABLES FOR FACTORIAL EXPERIMENTS

Source of variation	Sums of squares	Degrees of freedom	Mean squares	F	F(0.90)
A. Normalized Information Statistic (NIS)					
T	6.79	1	6.79	<1	
D	218.43	1	218.43	<1	
TD	8.20	1	8.20	<1	
M	1067.37	1	1067.37	4.52	2.96
TM	0.42	1	0.42	<1	
DM	63.87	1	63.87	<1	
TDM	176.36	1	176.36	<1	
Q	1055.93	3	351.97	1.49	
ERROR	4959.60	21	236.17		
TOTAL	7556.97	31			
B. Recall					
T	0.02832	1	0.02832	<1	
D	.24746	1	.24767	3.17	2.96
TD	.00720	1	.00720	<1	
M	.15318	1	.15318	1.96	
TM	.03920	1	.03920	<1	
DM	.00189	1	.00189	<1	
TDM	.07801	1	.07801	1.002	
Q	.50867	3	.16956	2.17	2.38
ERROR	1.63775	21	.07799		
TOTAL	2.70169	31			
C. Precision					
T	0.01565	1	0.01565	<1	
D	.00262	1	.00262	<1	
TD	.00902	1	.00902	<1	
M	.30574	1	.30574	8.48	2.96
TM	.02497	1	.02497	<1	
DM	.00017	1	.00017	<1	
TDM	.01374	1	.01374	<1	
Q	.08101	3	.02700	<1	
ERROR	.75667	21	.03602		
TOTAL	1.20959	31			

Confidence limits for the true difference $\Delta = [NIS(M_1) - NIS(M_2)]$ between the treatment means at the $(1 - \alpha)$ confidence level are given by:(105)

$$\left[\bar{\Delta} - t(\nu,\alpha/2)S_e\left(\frac{1}{r_1} + \frac{1}{r_2}\right)^{1/2}\right] \leq \delta \leq \left[\bar{\Delta} + t(\nu,\alpha/2)S_e\left(\frac{1}{r_1} + \frac{1}{r_2}\right)^{1/2}\right] \quad (8\text{-}1)$$

where

δ = the true difference in treatment means;

$\bar{\Delta}$ = the observed difference in treatment means;

S_e = the square root of the mean square due to error;

r_1, r_2 = the number of data points used to compute the treatment means;

α = the error probability; and

$t(\nu,\alpha/2)$ = the student's t statistic with ν degrees of freedom.

For the difference in NIS mean response we have $S_e = \sqrt{236.2} = 15.4$, $\alpha = 0.10$, $r_1 = r_2 = 16$, $\nu = 21$ and $t(21,0.05) = 1.721$. The 90 percent confidence interval for the true NIS difference is thus:

$$2.19 \leq [NIS(M_1) - NIS(M_2)] \leq 20.89.$$

Note that although $NIS(M_1)$ is estimated to be twice as large as $NIS(M_2)$ there is considerable room for improvement in M_1, since this method is operating only at 21.67 percent efficiency.

8.12 Dependence of Precision on Methods

The ANOVA table for search precision is shown in Table 8-1C. The only factor significantly affecting search precision is methods (M). The treatment means are:

$$\overline{P}(M_1) = 0.245$$

$$\overline{P}(M_2) = 0.050$$

$$\overline{\Delta} = \overline{P}(M_1) - \overline{P}(M_2) = 0.195 .$$

The difference $\overline{\Delta}$ is significant at the 99 percent confidence level. By using (8-1), a 99 percent confidence interval can be established for the <u>true</u> difference in search precision:

$$0.005 \leq [P(M_1) - P(M_2)] \leq 0.385.$$

For this application, $t(\nu,\alpha/2) \doteq t(21;0.005) = 2.83$, $S_e = \sqrt{0.036} = 0.190$ and $r_1 = r_2 = 16$.

The mean precisions given above are for individual searches. Comparing <u>pooled</u> M_1 and M_2 searches provides an illustration of the large difference in search precision. A total of 484 documents were predicted relevant by the 16 M_1 searches, and 50 of these were actually relevant. For the 16 M_2 searches, <u>1484</u> documents were predicted relevant, with 43 being actually relevant.

8.13 Dependence of Recall on Training Set Size

The ANOVA table for search recall is shown in Table 8-1B. Only the number of documents in the training set (factor D) significantly affects search recall. The training sets with the most documents lead to searches with better recall. The experiment treatment means are:

$$\overline{R}(D_1) = 0.264$$

$$\overline{R}(D_2) = 0.440$$

$$\overline{\Delta} = \overline{R}(D_2) - \overline{R}(D_1) = 0.176 \text{ .}$$

The 90 percent confidence interval for the true difference between treatment means is given below by (8-1) with $r_1 = r_2 = 16$, $t(\nu,\alpha/2) = t(21, 0.05) = 1.721$ and $S_e = \sqrt{0.078} = 0.279$:

$$0.07 \leq [R(D_2) - R(D_1)] \leq .345 \text{ .}$$

Comparing pooled D_1 and D_2 searches further illustrates the observed differences in search recall. A <u>perfect</u> retrieval system would have found 132 relevant documents for either the 16 D_1 searches or the 16 D_2 searches. In the experiment, the 16 D_1 searches found only 36 of these, while the 16 D_2 searches located 57 of them.

8.14 Lack of an Effect Due to the Number of BRS Index Terms

The number of BRS index terms (factor T) did not have a significant effect on either the search NIS, precision, or recall. This is somewhat unexpected, and may be due in part to an unfortunate source of uncontrolled variation in the experiment.

The nominal levels of factor T were set at 5 and 15 because these levels were approximate upper and lower limits for the number of index terms used normally by analysts in their BRS's. Accordingly, the machine system selected the 'best' 5 or 15 index term column vectors for inclusion in the L_1 approximation problem. Unfortunately, these chosen binary column vectors were not often linearly independent, and thus the optimal basis in the linear programming problem contains fewer than 5 or 15 index term vectors with non-zero weights. (For a further discussion of this, refer to section 5.6.) The final number of terms in the M_2 BRS's was correspondingly reduced to less than $T_1 = 5$ or $T_2 = 15$. This is illustrated by the data of Table 7-5, where the column labeled L shows the actual number of index terms appearing in the BRS. The average 'high' level (T_2) is 10.4 index terms (instead of 15), and the average low level (T_1) is 3.8 instead of 5.

Experimentally, this would have the effect of 'smearing' the level of factor T, and might mask effects of variation due to this factor. The levels of factor T in the experiment must be considered qualitatively a 'high' or 'low' instead of quantitatively as was originally intended. Suggestions are offered in section 5.6 for overcoming this difficulty in future applications by modifying the LP program.

8.15 Summary of the Factorial Experiment

The factor M (methods) had a significant effect on both the search precision and NIS. Furthermore, it was the only experimental factor which had an effect on precision or the NIS. The 16 M_1 searches (heuristic BRS's) had an average NIS response of 21.67 and an average precision of 0.245. The 16 M_2 searches (machine BRS's) had an average NIS of 10.13 and an average precision of 0.050. From Figure 7-1, virtually the entire observed average difference in NIS response between M_1 and M_2 can be attributed to the observed average difference in precision between M_1 and M_2. This large observed difference in average search precision between M_1 and M_2 is felt to be related to differences in selection of index terms and structural form of the BRS. Evidence for this will be presented in subsequent sections.

Search recall was observed to significantly depend on the number of documents in the training set, and to be independent of the search method. The average search recall for the 25 document training set (D_1) was 0.264, while the 50 document training set (D_2) led to searches with an average recall of 0.440.

The number of index terms (nominally $T_1 = 5$ and $T_2 = 15$) extracted from the training set and used for subsequent BRS formation had no observed significant effect on the search recall, precision or NIS. The levels T_1 and T_2 varied somewhat during experimentation. This may have helped to obscure a true effect if one were actually present.

8.2 Analysis of Variance for the Latin Square Sub-Experiment

This experiment, as discussed in section 7.37 was designed to determine whether there are significant differences between analysts, questions or treatments when method M_1 (heuristic BRS formation) is considered alone. Response data for this experiment appears in Table 7-2B. The ANOVA is shown below in Table 8-2.

Conclusions are simple. There are no significant effects attributable to either analysts, questions or treatments which are discernible from the experiment data at the chosen 90 percent confidence level (or even at the 75% confidence level).

TABLE 8-2. - ANALYSIS OF VARIANCE TABLE FOR LATIN SQUARE EXPERIMENT

Source of variation	Fixed or random	Expected mean squares	df.	SS	MS	F	F(0.75)
ANALYSTS(A)	R	$\sigma_e^2 + 16\sigma_A^2$	3	1396.98	465.66	1.58	1.78
QUESTIONS(Q)	R	$\sigma_e^2 + 16\sigma_Q^2$	3	837.90	279.30	<1	
TREATMENTS(T)	F	$\sigma_e^2 + 16\sigma_T^2$	3	394.85	131.62	<1	
ERROR	R	σ_e^2	6	1766.94	294.49		
TOTAL			15	4396.66			

8.3 Extraction of Best Index Terms

8.31 Distribution of $\hat{R}$

Table 7-4 was discussed in section 7.44. This table shows the relative frequencies of observed values of $\hat{R} = H(X) - H(X/Y)$ for the eight training sets which were used to generate the experimental BRS's.

The quantity $(0.693)(2N\hat{R})$ is asymptotically distributed as a chi-squared variate with one degree of freedom (when $\hat{R}$ is in bits), under the null hypothesis that $R = 0$. (See section 4.54.) If the alpha error is fixed at 0.05, this null hypothesis can be rejected when $\hat{R}$ is greater than 0.1105 for index terms in a 25 document training set (N=25), or when $\hat{R}$ is greater than 0.0504 for the 50 document training set (N=50). Index terms meeting the above criteria can be considered statistically significant predictors of document relevance at the 95 percent confidence level.

From Table 7-4, the average number of index terms having a statistically significant value of $\hat{R}$ at the 95 percent confidence level is eight terms for each 25 document training set and 15 terms for each 50 document training set. These averages are in line with the nominal values ($T_1 = 5$ and $T_2 = 15$) chosen for the experiment using another criterion. (See section 8.14.)

8.32 Differences in Index Term Selection between Methods

There are two major differences between index terms selected using M_1 and M_2. These are: differences in $\hat{R}$ evaluated over the training set; and differences in the annual frequency of index term use.

8.321 Differences in $\hat{R}$. The individual index terms selected for the BRS using M_2 (machine methods) are those having the highest values of $\hat{R}$. The average value of $\hat{R}$ for index terms extracted heuristically (M_1) was only about half that of the average $\hat{R}$ using M_2. The overall search effectiveness (NIS), however, is better for M_1 than

M_2. It follows that the index terms chosen by analysts are better indicators of document relevancy over the file as a whole than are those selected by the M_2 machine methods. This suggests the use of extra information by analysts from outside the training set during the term selection process.

8.322 Differences in Frequencies of Term Occurrence. The frequency of term occurrence over the file as a whole was not a selection factor for method M_2 (machine). The annual frequency of occurrence for the M_2 index terms has a mean of 773 and a variance of 597,100. For method M_1, the population of index terms selected by analysts and used to construct BRS families with $S = 1$ (see section 7.45) has a mean annual frequency of occurrence of 177 and a variance of 31,300. The hypothesis that the mean frequencies of occurrence are the same for M_1 and M_2 index terms can be rejected at the 99.5 percent confidence level. This implies that the analysts of M_1 are utilizing frequency of occurrence information (which is not available from the training set) when they choose index terms. To summarize, the M_1 analysts select terms to use in their BRS's which have a frequency of occurrence lower by a factor of $773/177 \cong 4.37$ than those terms selected for the BRS's of method M_2.

8.33 The Sampling Problem

The problem of choosing a representative training set is one of sampling from the document file. A random sample is usually assumed for the training sets of pattern recognition systems. However, in a large document retrieval system, a randomly chosen sample for the training set is infeasible for practical reasons. To illustrate assume 500,000 documents are in a file, and that 100 of them are relevant. Now it would require, on the average, a random sample of 5,000 documents from this file to provide a training set which would include one relevant document. Clearly, a sample of this size is unmanageable. A training set with only 25 to 50 documents is considered typical. Some reasonable percentage (near half) of all training set documents should probably be relevant to insure reasonable retrieval results. Thus a typical training set with 50 documents (and 25 relevant) constitutes a highly enriched sample, as opposed to a random chosen training set.

The Results of section 8.32 indicate that the analysts of M_1 are using supplementary information to select index terms. It is interesting to relate this observation to the phenomenon of non-random sampling discussed above.

The data presented in sections 8.321 and 8.322 suggests that the supplementary information is of two forms. First, the analysts' knowledge of term occurrence frequency is used to avoid those terms which occur frequently, even though they have a high value of $\hat{R}$ over the enriched training set. Perhaps the analyst 'feels' (for example) that there are only 15 relevant documents in a one-month section of the file. This leads him to reject any index terms which he knows have more than 50 associated documents (on the average) in a one-month section of the file. If the training set size were greatly increased, it is felt that the same low frequency index terms would also be selected by method M_2.

Secondly, pure sampling error of a random nature may cause terms to appear to be good discriminators, when in fact, with a larger training set they would not be. These terms are excluded by the M_1 analysts because they do not 'fit in' with the analyst's concept of the query. Here the analysts supply information based on their prior knowledge of the query and their prior knowledge of language use.

In conclusion it is hypothesized that the supplementary information used by the analysts of M_1 to select index terms compensates for the small size and non-randomness (enrichment) of the training set. A high index term frequency of occurrence would tend to reduce the value of $\hat{R}$ for this term in M_2 if the sample size were increased. Also, the probability of observing unrelated index terms with a high $\hat{R}$, decreases as N, the sample size increases.

8.4 Analysis of the BRS

8.41 Dependence of BRS Solution Family Size on Methods

Table 7-5 shows the SIZE=S distribution of constituent families of the BRS's for all the experimental searches (see section 7.45). There are several striking differences between the BRS's for M_1 and M_2 when they are compared using the SIZE(S) of their constituent solution families. Table 8-3 presents this comparison.

(A) The M_1 analysts use (on the average) 27.2 solution families to make up a BRS (recall that each solution family is a 'matching template'). On the other hand, each M_2 BRS is composed of an average of 108.2 solution families.

(B) The M_1 analysts composed their BRS's using <u>only</u> solution families having $S \leq 2$. Of 435 solution families, only 15 (or 3.45%) had solution families with $S = 1$. For the M_2 BRS's, solution families with $S \leq 10$ were observed, with $S = 5$ being the most likely value. There were 124 (out of 1515) families with $S = 2$ (or 8.19%) and 36 with $S = 1$ (or 2.38%).

Because the M_1 analysts used <u>fewer</u> solution families per BRS, the <u>number</u> of M_1 solution families with $S = 1$ is less per BRS than the M_2 families with $S = 1$ (0.94 versus 2.57). This causes the total number of documents retrieved per BRS to be less (on the average) for M_1 than M_2.

TABLE 8-3. - COMPARISON OF BRS SOLUTION FAMILY SIZES FOR M_1 AND M_2

	M_1		M_2	
Solution family size	Average number per BRS	Average percent of BRS	Average number per BRS	Average percent of BRS
S=1	0.94	3.45	2.57	2.38
S=2	26.26	96.55	8.86	8.19
S≥3	------	------	96.77	89.43
Total	27.20	100.00	108.20	100.00

8.42 Effects of BRS Family Size on Retrieval System Operation

The SIZE=S of the solution families making up the BRS has an effect on retrieval system operation. The expected number of documents which a given family (or template) will match decreases as S increases. With $S=1$, only one term in a document is required to match the solution family. Thus, the expected number of matching documents in a file covering a given time span is simply the total number of documents indexed with the term in that time span. When $S=2$, all matching documents are required to have a pair of matching terms. One would expect (on the average) less documents to match a family with $S=2$ than with $S=1$.

The following approximate model is useful for descriptive purposes. Let $p \ll 1$ be the average probability that any given index term will be used to index a document. Then $q = 1 - p$ is the probability that a given term will not be used to index a given document. This assumes all terms are independent.

Consider a solution family F which has S variables fixed at 1, ℓ variables fixed at 0 and the rest arbitrary. Then, the probability of matching the given term combination in the family with a combination of terms in a document is $p(F)=p^S q^\ell \cong p^S$, since $q = 1 - p \cong 1$. For a file with N documents, there will be (on the average) $M = Np(F) = Np^S$ documents matching the solution family F. Now, by using $\log p \cong -\frac{1}{p}$ (since $p \cong 0$) in the expression $\log M = \log N + s \log p$

we have:

$$M = Ne^{-s/p} . \qquad (8\text{-}2)$$

To a rough approximation then, the number of documents matching (or retrieved by) a given BRS solution family decreases exponentially as the SIZE=S of the family increases.

By minimizing the use of solution families with $S=1$, the analysts of M_1 have cut down drastically on the number of documents which will be retrieved by the BRS. This should increase the M_1 search precision. By avoiding the use of families with $S \geq 3$ they have cut down the search costs by neglecting those documents which have a very low probability of matching the BRS.

8.5 Predicted Utilities of Relevant Documents for M_2

8.51 Factors Affecting the Recall of the M_2 System

Table 7-3B (discussed in section 7.43) shows that for the known relevant documents (with $(u - \tau) \geq 0$), 33.4 percent were correctly predicted to be relevant by the system (had $(\hat{u} - \tau) \geq 0$), 18.1 percent were incorrectly predicted to be non-relevant (had $(\hat{u} - \tau) < 0$), and 48.5 percent were missed because they had no index terms in common with index terms in the BRS. This data shows how the recall of the M_2 system is affected by errors, since only the relevant documents are analyzed.

The system made errors with 66.6 percent of the relevant documents. Of these 18.1/66.6 = 27.2 percent were misclassified by the LUPF and 48.5/66.6 = 72.8 percent were eliminated by the feature extraction process. This indicates that the feature extraction process very critically affects the M_2 system recall. Improvements in M_2 recall are most likely to be brought about by efforts to improve the feature extraction process instead of the LUPF estimation process.

8.52 Effects of Increasing the Vocabulary Size

Although not directly supported by data here, the vocabulary size (or number of index terms in the system master list) would seem to have an effect on the number of documents having no terms in common with the BRS. Some conjectures are made below.

As index terms are added to the master list, all relevant documents associated with a given query would show (on the average) less overlap in their index term sets. This implies that the relevant document index terms would also have less overlap with a 'best' BRS of given size. (It is assumed that indexing remains at a constant quality level, that the same number of index terms are assigned to a document before and after the master list is expanded, and that the method of BRS formation remains the same.) The reason for this is simply that there would be more terms for an indexer to choose from and hence the average frequency of individual term use would be reduced, assuming a constant file size.

As the term master list is reduced in size, term 'overlap' in the set of relevant documents should become greater. This would cause fewer relevant documents to be missed but more unrelated documents to be retrieved by a 'best' BRS of fixed size. This is because the terms and term combinations would be less specific with a reduced vocabulary. Stated another way, decreasing the vocabulary size should increase recall and decrease precision.

8.6 Summary of the Data Analysis

Many aspects of the experimental data have been analyzed in this chapter. Only the results which are felt to be most important are reviewed here.

From section 8-1 it is concluded that search effectiveness (in terms of the NIS) is significantly greater for method M_1 (analysts) than for method M_2 (machine). It is shown that this difference can be attributed wholly to the significant differences in search precision between M_1 and M_2. In other words, M_1 and M_2 recover nearly the same fraction of relevant documents (recall is the same), but method M_2 retrieves many more non-relevant documents (a lower search precision).

Section 8.3 shows that index terms selected by analysts differ significantly from those selected by machine methods. The major difference is that the M_2 terms have a much higher frequency of occurrence. This is undesirable, since it causes more documents to be re trieved, which reduces M_2 search precision. By using supplementary information about term occurrence, the analysts are apparently able to eliminate index terms which would have been eliminated had the training sets been randomly chosen, and hence been much larger.

Section 8.4 demonstrates that the index term chosen by the M_1 analysts are combined in a much different manner (to form a BRS) than are the M_2 terms. In particular, a greater number of solution families appear in the M_2 BRS's. Also, the M_2 BRS's are constructed largely of solution families with $S \geq 3$, while for M_1, nearly all families have $S = 2$. Families with $S = 1$ appear an average of 2.57 times per BRS with M_2, and only 0.94 times per BRS with M_1.

The selection of terms with a low frequency of occurrence, together with the avoidance of solution families with $S = 1$ constitute the major differences between M_1 and M_2. These two differences working jointly would account for large differences in search precision between M_1 and M_2. It appears that any attempt to make the machine method M_2 comparable with M_1 will have to resolve these differences.

Section 8.5 analyzes errors which reduced the M_2 search recall. About 73 percent of the relevant documents were missed because they had no index terms in common with the BRS. This indicates again that improvements in the term selection process would have a major effect on search effectiveness.

8.7 Conclusions

The results of the experimentation illustrate the basic applicability of pattern recognition techniques to the document retrieval problem.

Test results conclusively show the superiority of the analysts to the machine recognition system developed here. The clear superiority of humans to machine systems for recognition of visual patterns is well known. It is one of the reasons for the enduring academic interest in pattern recognition processes. Thus it is not surprising that patterns consisting of index terms should be recognized more efficiently by humans than by machine methods.

What is surprising and encouraging is that the resolution of the current differences in system effectiveness does not appear to be out of the realm of possibility. The current best estimated difference of 11.5 percent in the NIS can possibly be resolved by extending and refining the model. In particular, two refinements are felt to be most promising.

First, the methods of index term selection should be extended to incorporate term frequency of occurrence information. This would tend to compensate for the non-randomness of the training or sample set.

Secondly, restrictions should be placed on the BRS to reduce the number of solution families with $S = 1$ and $S \geq 3$.

The above refinements are discussed in chapter 9. They both should improve the search <u>precision</u> of M_2 relative to M_1, and make the differences in overall effectiveness less for the two methods. A number of other reasonable extensions to the present M_2 system are also mentioned in chapter 9.

9.0 SUGGESTIONS FOR FURTHER RESEARCH

9.1 General

Several suggestions for further research can be made as a result of this study. These can be more or less divided into five distinct areas, which are summarized very briefly below before details are given.

(A) The information statistic for selecting index terms can be modified to take term frequency of occurrence into account.

(B) Instead of selecting the best <u>single</u> index terms; term <u>pairs</u> or <u>triplets</u>, etc., can be selected which have a high information content over the training set. This is a form of higher order feature extraction.

(C) The approximation theory model can be altered. Possible modifications include a change of norm from L_1 to L_∞ or L_2; use of {0,1,2} variables for x_{ij} based on 'major' or 'minor' terms in the training set; use of rougher utility estimates (say +1 or -1) for documents in the training set; and secondary selection of alternate optimal solutions based on frequency of occurrence of index terms. Also, alternate algorithms can be investigated for more efficient solution of the approximation problem.

(D) The solutions of the LPBI can be constrained so that only solution families with $S \leq 2$ or $S = 2$ will be found. This is easily done by solving a two-inequality <u>system</u> instead of a single inequality.

(E) The important effect of iterative improvement of the training set by repeated searches of the file can be considered as an extension of the previous test methods.

9.2 Modifications to the Information Statistic for Selecting Index Terms

9.21 Incorporating Information about Frequency of Term Occurrence

A revised measure of goodness for index term selection which utilizes index term frequency of occurrence information is desired. One such measure would be $(\hat{R}_j/f_j)$ which would replace $(\hat{R}_j)$. Here f_j is the expected frequency of occurrence of term j over the section of file to be searched. This measure would reduce the estimated effectiveness $\hat{R}_j$ of the individual term if it occurred very frequently. For example, the term 'computer·program' might be judged excellent based on the training set value of $\hat{R}$, but knowing that it occurred 1000 times per year might change this judgment. This would be especially true if a prior user estimate were available to the effect that no more than 50 documents were relevant in the annual file.

9.22 Utilizing More Refined Document Utility Measurements

It is also possible to derive a more refined $\hat{R}$ without using information about frequency of term occurrence. The present scheme assumes a binary utility measure (relevant or not relevant), and derives the information statistic from the 2 x 2 contingency table shown below. The entries in the table are obtained from the training set.

	Term present	Term absent		
Relevant $(u \geq \tau)$	n_{11}	n_{12}	$n_{1\cdot}$	(9-1)
Not relevant $(u < \tau)$	n_{21}	n_{22}	$n_{2\cdot}$	
	$n_{\cdot 1}$	$n_{\cdot 2}$	N	

Since more refined utility measures are available, a more extensive table could be set up as shown below:

	Term present	Term absent		
$u = 1$	n_{11}	n_{12}	$n_{2\cdot}$	
$u = 2$	n_{21}	n_{22}	$n_{2\cdot}$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	(9-2)
$u = 9$	n_{91}	n_{92}	$n_{9\cdot}$	
	$n_{\cdot 1}$	$n_{\cdot 2}$	N	

Table (9-2) can be used instead of (9-1) to determine $\hat{R} = H(X) - H(X/Y)$ by direct calculation.

9.3 Applying the Feature Selection Process to Different Types of Features

9.31 Higher Order Features

Either single index terms or term combinations can be considered as pattern 'features'. The system tested extracted the best single-term features. It is possible to consider other types of index term 'features'. For example, all training set index terms can be arranged in paris (T_i, T_j), triplets (T_i, T_j, T_k), etc., having <u>fixed configurations</u>. Any one of these fixed configurations can be considered as a binary 'feature' and an information statistic $\hat{R}$ can be derived for it.

For an example of two-term features, consider the term pair (T_i, T_j). There are <u>four</u> fixed configurations in which to arrange this pair of terms, i.e.

$$(T_i \cap T_j) = (T_i T_j)$$

$$(T_i \cap \overline{T}_j) = (T_i \overline{T}_j)$$

$$(\overline{T}_i \cap T_j) = (\overline{T}_i T_j)$$

$$(\overline{T}_i \cap \overline{T}_j) = (\overline{T}_i \overline{T}_j)$$

Since the same information is contained in $(T_i T_j)$ as is contained in $(\overline{T}_i \overline{T}_j)$, there are only three <u>different fixed configurations</u> to consider. For a training set with 200 different terms, there would be

$3\binom{200}{2} = 3(19{,}900) = 59{,}700$ term-pair features to consider individually. Each of these features would require a corresponding $\hat{R}$ computation.

Methods to avoid complete enumeration when searching for good term-pair features have been discussed by Swonger[106]. If the 'features' extracted are of the multiple index term type, the LUPF will be of the form $\sum \gamma_j f_j = y_i$, where f_j are features such as $(T_1 \overline{T}_3)$. When this LUPF is thresholded, the resulting pseudo-Boolean inequality is no longer linear. Luckily, solving a non-linear pseudo-Boolean inequality can be accomplished as a simple extension of the linear theory. This will be discussed in section 9.42.

9.32 Selection of Features for Training Set Coverage

The results of section 8.51 showed that 48.5 percent of the relevant documents were missed because they had no terms in common with those in the set of selected index terms. This suggests that perhaps single-term features or term-pair features be chosen not only for their good discrimination qualities, but also for their degree of 'coverage' of the training set. One way of insuring better coverage is to choose features with high information statistics, but with low pairwise correlation coefficients. This type of correlation screening has been studied by Maltz[107] for binary features extracted from two-dimensional patterns.

9.33 Major and Minor Index Terms in the NASA System

All index terms occurring in the NASA system are assigned as either 'major' or 'minor' terms. Major terms are intended to indicate major concepts in the document, while minor terms are used in a supporting role. Selecting only from the set of major terms would be one way of utilizing this built-in form of feature extraction.

9.4 Modifications to the BRS Structure

9.41 Avoiding Solution Families with $S = 1$

By changing the structure of the BRS to avoid solution families with $S = 1$, the precision of the search may be increased. One way of doing this is to incorporate constraints directly on the binary variables of the LUPF. For example, to restrict the SIZE of all solution families to be less than or equal to 2, we can solve the system given by

$$\sum_j a_j T_j \geq (\tau - a_0)$$

$$\sum_j T_j \leq 2$$

Another, more indirect way of restricting the use of frequently occurring index terms would be to solve a system such as the following:

$$\sum_j a_j T_j \geq (\tau - a_o)$$

$$\sum_j f_j T_j \leq N$$

where N is the maximum (expected) number of documents desired per time period and the f_j are expected frequencies of term occurrence for the same time period.

Methods for solving systems of linear pseudo-Boolean inequalities are discussed by Hammer and Rudeanu.[108]

9.42 Solving the Nonlinear Pseudo-Boolean Inequality

As mentioned in section 9.31, choice of other than single-term features leads to a pseudo-Boolean inequality which has the form

$$\sum_{j=1}^{n} a_j f_j \geq (\tau - a_o) .$$

As an example, consider

$$a_1(T_1\overline{T}_2T_5) + a_2(T_2T_3) + a_3(\overline{T}_1\overline{T}_4T_5) \geq \tau - a_o .$$

This nonlinear inequality may be solved by using simple extensions of the methods used for linear inequalities in chapter 6. See, for instance, Hammer and Rudeanu[109]. To solve the nonlinear inequality, define new binary variables y_j:

$$y_1 = T_1\overline{T}_2T_5$$

$$y_2 = T_2T_3$$

$$y_3 = \overline{T}_1\overline{T}_4T_5 \ .$$

Then solve the linear inequality given by

$$\sum_{j=1}^{3} a_j y_j \geq (\tau - a_o) \ .$$

After the m solution families $F_K(\underline{y})$; $K = 1,2,\cdots,m$ are obtained for this linear inequality, the original variables are substituted into the expressions for the linear families $F_K(\underline{y})$ as follows

$$y_j \leftarrow f_j$$

$$\Longrightarrow F_K(\underline{y}) \leftarrow F_K(\underline{T}) .$$

Finally, after simplifying the resulting expressions for $F_K(\underline{T})$, we have the desired solution families for the nonlinear inequality. Thus the specification of multi-term features does not introduce severe computational difficulties.

9.5 Derivation of the LUPF

Several modifications and extensions are discussed below, all of which retain the linear model for predicting document utility.

9.51 Choice of Norm

Parameters in the LUPF could be estimated from the training set by using the minimal value of the L_∞ or L_2 norm as a measure of goodness of fit instead of the minimal L_1 norm. The L_∞ problem also has a formulation as a linear programming problem[110,111].

9.52 Selection Among Alternate Optimal Solutions

Both L_1 and L_∞ problems suffer from the 'disadvantage' of admitting alternate optimal solutions. This could be used to advantage by selecting among alternate optimal solutions as a post-optimal procedure. A secondary function based on frequency of term occurrence could be used for this purpose.

9.53 Choice of Independent Variables

The choice of independent variables x_{ij} was very simple for the problem tested. Here $x_{ij} \varepsilon \{0,1\}$ depending on whether or not a feature (term) j is present with document i. A simple extension is to let $x_{ij} \varepsilon \{0,1,2\}$ where now $x_{ij} = 1$ if term j is a <u>minor</u> term with document j and $x_{ij} = 2$ if term j is a <u>major</u> term. (See section 9.33.)

When the LUPF (formed using $x_{ij} \varepsilon \{0,1,2\}$) is thresholded, it no longer gives a pseudo-Boolean inequality. This difficulty can be overcome by <u>converting</u> the <u>integer</u> inequality to an equivalent <u>system</u> of pseudo-Boolean inequalities. See, for instance, Hammer and Rudeanu[112].

9.54 Choice of Dependent Variables

The dependent variable y_i is document utility. In the test configuration $y_i \varepsilon \{1,2,\cdots 9\}$. A much simpler form and one which might work just as well would be to let $y_i \varepsilon \{-1,+1\}$ as a measure of relevance for documents in the training set. Then a value of $\tau = 0$ could be used to form the Boolean inequality.

9.55 LP Problems with Unequal Slack Costs

With the L_1 approximation problem formulated as a linear programming problem, the initial basis is composed entirely of slack vectors. As these slack vectors are driven out of the basis the L_1 norm is minimized. When each slack vector has unit weight (or cost) in the objective function, there is no preference given to one slack vector over another. Each has an equal opportunity to be driven from the basis. Every slack vector is associated with one row of the constraint matrix, which represents a single document in the training set. When a slack vector is driven out of the basis, the residual for this row drops to zero and a perfect fit to the predicted document utility is realized.

By assigning <u>different</u> objective function weights to slack vectors, it is possible to force a better fit to the part of the training set with the higher weights, at the expense of the part of the training set with the lower weights. This can be used in at least two ways.

9.551 Forced Fitting to the Relevant Documents. By assigning higher weights to slack associated with the training set which are relevant, and lower weights to those documents which are non-relevant, the utilities of the relevant documents will be fit at the expense of the non-relevant ones. This may result in improved search quality.

9.552 Application to Iterative Retrieval. With iterative retrieval the training set grows in size following repeated retrieval efforts on the same file. Consider an exponential decrease in the weights of slack vectors corresponding to sample documents according to the time which they have remained in the training set (i.e., $w_j = e^{-\tau_j n}$ for the n^{th} time in the training set). The relative importance of training set documents decreases as they become 'older'. Thus, the older documents are gradually 'forgotten', and the LUPF derived is more closely tuned to the most recently acquired members of the training set. This is one way to effectively limit the size of a large training set, and also to following the changing interests of a user.

9.56 Improved Algorithms

While only marginally related to the document retrieval problem, more efficient methods of solving the L_1 approximation problem are suggested by the nature of the basis inverses arising from the LP problem. In particular, it has been observed that elements of the basis inverses are integral multiples of integral powers of 1/2 when the document utilities are specified as positive integers. The LP solution variables have been observed to be integral multiples of 1/2.

9.6 Experimental Investigation of Iterative Retrieval

The ability of a document retrieval system to adapt to changing user needs has become especially important with the advent of time-sharing search systems which allow rapid implementation of successive BRS's.

The system tested in this dissertation has been of the 'static', _single_ search type. In an iterative configuration the same file would be _repeatedly_ searched a number of times, with modifications being made to the training set after each search. Following a sequence of searches, it is hypothesized that an asymptotic level of search effectiveness would be reached, which would be significantly greater than that of a 'single search' system.

Test methods for use with an iterative configuration could be the _same_ as those employed for the testing here, except for two complications. First, rules regarding additions and deletions to the training set would have to be established. Perhaps the size of the training set would be limited, with new additions forcing an equal number of deletions. Alternately, the training set size could be unrestricted, and the 'older' documents 'forgotten' as outlined in section 9.552. Secondly, a stopping rule would have to be imposed to restrict the number of iterations. This could be simply a limit on the allowable number of searches. The effectiveness of the _final_ search could become the dependent variable, instead of the effectiveness of the _only_ search as was done here.

APPENDIX A - AN EXAMPLE PROBLEM

To provide an overview of system operation, the solution of a representative problem is presented here. A training set of pattern vectors (representing documents having user assigned utilities) is processed. First, index terms are selected in a feature extraction operation. This is followed by solving an L_1 approximation problem for document utility as a function of index term 'weights'. Finally, the LUPF is thresholded to give an LPBI. This is solved for solution families (index term matching templates). The union of these templates is a BRS. Results are illustrated with actual computer output. The system has been programmed in Fortran IV for the IBM 7094/7044 Direct Couple System.

A.1 Input Data

The input data to process a 28 document training set is shown on Figs. A-1 to A-4. The first card read in (not shown) gives the number of documents in the training set (28) and the utility threshold ($\tau = 3$) which defines relevancy on the scale of 1-9 (integer) used to rate all documents in the training set. A document is considered relevant if its utility is greater than or equal to 3 and not relevant otherwise.

For each document in the training set, the following items are read in:

(a) document number (treated as an alphanumeric character string);

(b) number of index terms;

(c) user assigned utility; and

(d) actual index terms (also treated as alphanumeric character strings).

The training set documents are processed in sequential order. Each document number is read and stored as a character string and assigned a new number (an integer) which is used by the program for further processing. Fig. A-5 shows the document data summary.

A.2 Processing of Index Terms

Figures A-6 to A-8 show an alphabetical listing of all index terms occurring in the training set and their associated information statistics (see chapter 4). Each index term is read in and stored as a character string but for all further processing is represented by an internal index term number (an integer). A total of 155 index terms were found with the 28 documents of the training set.

Figures A-9 to A-11 show the same list of index terms sorted on their information statistics instead of alphabetically. (The larger the the information statistic, the more effective the index term is at discriminating between relevant and nonrelevant documents.)

A.3 The Document-Term Matrix and Computation of $\hat{R}$

Figures A-12 to A-14 show the document-term matrix which it will be convenient to denote as $T = (t_{ij})$. Each row corresponds to an index term and each column represents a document in the training set. If index term i appears in document j, then $t_{ij} = 1$; otherwise $t_{ij} = 0$. At the top of Fig. A-12 the document utilities are shown over the document category designation (1 for a relevant document, 0 otherwise). This category vector is formed by applying the utility threshold $\tau = 3$ to the document utilities.

To compute the information statistics, the 0/1 row vector in T for each index term is compared with the 0/1 category vector in a 2 x 2 contingency table. The information statistic $\hat{R}$ is a measure of the <u>similarity</u> of the two vectors.

A.4 Solving the L_1 Norm Approximation Problem

Index term weights are determined by solving a linear approximation problem using the L_1 norm as the criterion of goodness. This approximation problem is set up as a linear programming problem and solved using the simplex algorithm (see chapter 5). Prior to solving the problem, all index terms are discarded except those ten having the highest information statistics. Only these ten terms appear in the approximation problem. They represent extracted features and are used to best approximate assigned document utilities as a linear combination of term weights. The linear programming problem has the following form:

$$\text{minimize} \quad z = \underline{c}'\underline{x}$$

$$\text{subject to} \quad A\underline{x} = b$$

$$\text{and} \quad \underline{x} \geq \underline{o} .$$

Figures A-15 to A-17 show the matrix A and the vectors $\underline{b}$ and $\underline{c}$ which result from setting up the approximation problem using only the ten best terms. There are 28 rows in the matrix A and 78 columns. Data is listed by columns. (A(13,6) for example is -1.00). Cost data (c_j) are listed with each matrix column. All costs are either 0 (non-slack cols. 1-22) or 1 (slack cols. 23-78). The right hand side ($\underline{b}$) is shown in Fig. A-17.

The elements of the right hand side vector $\underline{b} = (b_i)$ are the utilities assigned to the documents. The first eleven columns of the matrix A (I,J) correspond to a constant a_o (first column) plus the 0/1 vectors from the document term matrix corresponding to the ten index terms with the largest information statistics.

Figure A-18 shows a solution summary printed after the linear programming problem was solved. This figure relates the basic variable numbers (structural columns in the optimal basis) to the actual index terms and the slack variables.

The value of the objective function is the length of the residual vector in the L_1 sense (that length is $L_1 = 7$ in the problem solved here). Based on data shown here, the best L_1 LUPF is:

$$\hat{u} = a_o + \sum_{j=1}^{7} a_j T_j \qquad \text{(A-1)}$$

$$= 1.0 + 4.0T_1 - 4.5T_2 + 4.0T_3 + 2.0T_4 + 5.5T_5 + 2.5T_6 + 3.0T_7 .$$

where $\hat{u}$ is the predicted utility, $a_o = 1.0$ is the constant term weight and a_j are the weights for index terms 1 to 7. Although the L_1 approximation problem was set up to determine weights of ten terms, only seven terms have non-zero weight in the optimal solution. This phenomenon is discussed in chapter 5. It occurs because of linearly dependent index term columns in the original structural matrix. Fig. A-19 shows a computation of residuals using the derived utility prediction equation. A comparison can easily be made between the user assigned document utilities and the utilities predicted by the linear model. For example, document ten has an assigned utility of four and a predicted utility of three.

A.5 Solving the LPBI

The LUPF derived previously can now be thresholded to give an LPBI (see chapter 6). Using the threshold $\tau = 3$ read in with the data, we get

$$4.0T_1 - 4.5T_2 + 4.0T_3 + 2.0T_4 + 5.5T_5 + 2.5T_6 + 3.0T_7 \geq 2.0 . \qquad \text{(A-2)}$$

Before this LPBI can be solved, it is necessary to convert all coefficients to integers. Multiplying the inequality by 10 gives

$$40T_1 - 45T_2 + 40T_3 + 20T_4 + 55T_5 + 25T_6 + 30T_7 \geq 20 \ . \quad \text{(A-3)}$$

These data are summarized in Fig. A-20.

(The notation used in the program here to describe the parameters of the LPBI (A-3) on Fig. A-20 is slightly different than that used in chapter 6. The exponents α_j given in (6-2) are referred to as COMPLEMENT(J) in the program here. Also, when $\alpha_j = 1$, COMPLEMENT (J) = 0.)

The next step in the solution of the LPBI is to convert it to canonical form (see chapter 6). This form has no negative coefficients, and all coefficients are sorted according to magnitude. The coefficients of the canonical form are also shown in Fig. A-20.

The branch-and-exclude algorithm described in chapter 6 gives 17 basic solutions to the canonical form. These are shown in Fig. A-21A.

The basic solutions are converted to canonical families of solutions and then transformed back to their original (non-canonical) form. The 17 non-canonical families of solutions are shown on Fig. A-21B. Each solution family represents a Boolean template of index terms which can be used for retrieving from an inverted file. The 1's are interpreted as the required presence of a term, the 0's indicate the required absence of a term and the 2's indicate indifference as to whether the term is present or absent. The 1's and 0's correspond to fixed variables, while the 2's correspond to free or arbitrary variables. For example, solution family 12 specifies the retrieval of all documents which have term 5 present and term 2 absent, and with indifference as to whether terms 1,3,4,6,7 are present or not. The complete BRS is given by the union of all solution families.

A.6 Miscellaneous Results

Near the right margin of the page on Fig. A-21B are shown the variables MIN, BASE, MAX and SIZE, which pertain to each of the solution families listed near the left margin of Fig. A-21B. The variables MIN, BASE and MAX are related to the range of predicted utilities associated with each of the solution families. (See Section 6.73) The following terminology is introduced to describe this relationship.

We are given the LPBI from the linear programming solution (A-2):

$$\sum_{j=1}^{n} a_j T_j \geq (\tau - a_o) . \tag{A-4}$$

We multiply this inequality by the appropriate constant γ, giving a new inequality (A-3) with integer coefficients:

$$\sum_{j=1}^{n} a_j^* T_j \geq (\tau^* - a_o^*)$$

where $\quad a_j^* = \gamma a_j , \ j=0,1,2,\cdots,n \qquad$ (A-5)

and $\quad \tau^* = \gamma\tau$.

(In the sample problem; $\gamma = 10$, $\tau^* = 30$, $n = 7$ and $a_o^* = 10$ from (A-1) through (A-3).) Next, we solve this inequality for its M families of solutions $F_k(\underline{T})$, $k=1,2,\cdots,M$. (In the example problem, $M = 17$.)

Designate the set of fixed indices j associated with the k^{th} solution family as S_{k1} and the set of free indices as S_{k2}. (For example with $k = 12$; $S_{k1} = \{2,5\}$ and $S_{k2} = \{1,3,4,6,7\}$.) Now define for each family k the following:

$$\mathrm{BASE}(k) = \sum_{j \in S_{k1}} a_j^* T_j ; \tag{A-6}$$

$$\mathrm{MAX}(k) = \max_{S_{k2}} \left[\sum_{j=1}^{n} a_j^* T_j \right] ; \tag{A-7}$$

and

$$\mathrm{MIN}(k) = \min_{S_{k2}} \left[\sum_{j=1}^{n} a_j^* T_j \right] . \tag{A-8}$$

(For the sample problem, BASE(12) = 55, MAX(12) = 210 and MIN(12) = 55, as shown on Fig. A-21B.)

Quantities (A-6) through (A-8) can be related to the end points of the range of predicted utility $\hat{u}(k)$ for the k^{th} solution family of the original inequality (A-1) as follows:

$$\min \hat{u}(k) = \min_{j\varepsilon S_{k2}} \left[a_o + \sum_{j=1}^{n} a_j T_j \right]$$

$$= \frac{1}{\gamma} \left[a_o^* + \min_{j\varepsilon S_{k2}} \left(\sum_{j=1}^{n} a_j^* T_j \right) \right] = \frac{1}{\gamma} [a_o^* + \mathrm{MIN}(k)]; \qquad \text{(A-9)}$$

and $$\max \hat{u}(k) = \max_{j\varepsilon S_{k2}} \left[a_o + \sum_{j=1}^{n} a_j T_j \right]$$

$$= \frac{1}{\gamma} \left[a_o^* + \max_{j\varepsilon S_{k2}} \left(\sum_{j=1}^{n} a_j^* T_j \right) \right] = \frac{1}{\gamma} [a_o^* + \mathrm{MAX}(k)]. \quad \text{(A-10)}$$

For the sample problem, using (A-9) and (A-10) gives:

$$\min \hat{u}(12) = \frac{1}{10} [10 + 55] = 6.5 \qquad \text{(A-11)}$$

and $$\max \hat{u}(12) = \frac{1}{10} [10 + 210] = 22.$$

Thus we have $6.5 \leq \hat{u}(12) \leq 22$. In a similar manner ranges of predicted utility can be established for each of the solution families shown in Fig. A-21B by using (A-9), (A-10) and the given data.

BASE(k) is used as a preliminary result in the computation of MIN(k) and MAX(k). To illustrate this, consider

$$\mathrm{MAX}(k) = \max_{S_{k2}} \left[\sum_{j=1}^{n} a_j^* T_j \right]$$

$$= \max_{S_{k2}} \left[\sum_{j \varepsilon S_{k1}} a_j^* T_j + \sum_{j \varepsilon S_{k2}} a_j^* T_j \right]$$

$$= \sum_{j \varepsilon S_{k1}} a_j^* T_j + \max_{S_{k2}} \left[\sum_{j \varepsilon S_{k2}} a_j^* T_j \right]$$

$$= \mathrm{BASE}(k) + \max_{S_{k2}} \left[\sum_{j \varepsilon S_{k2}} a_j^* T_j \right] . \qquad \text{(A-12)}$$

A similar result holds for MIN(k).

The SIZE of a solution family is defined as the number of 1's in it. This variable is shown on Fig. A-21. Each 1 specifies the required <u>presence</u> of an index term in any document vector which would match the family (or template). Very roughly, the probability P of finding a document which matches a given template is given by (see section 8.42)

$$P(\mathrm{match}) = e^{-s/p} \qquad \text{(A-13)}$$

where p is the average probability that an index term will be used, and s is the SIZE of the family. The larger the SIZE of a solution family, the greater are the chances that no documents will be found which will match it.

Each solution family has the pleasant property that any document retrieved using it will <u>not</u> be retrieved by any other reduced solution family. This can be verified by noting that each solution family of Fig. A-21 differs from the others by at least one 1 being changed to 0 or vice versa.

DOCUMENT NUMBER	INDEX TERM COUNT	DOCUMENT UTILITY
68N10674	10	07

BIBLIOGRAPHIES
CONTAMINANTS
MICROWAVE SPECTRA
MOLECULAR STRUCTURE
SPACECRAFT CABIN ATMOSPHERES
CHEMICAL ANALYSIS
INORGANIC COMPOUNDS
MOLECULAR SPECTROSCOPY
ORGANIC COMPOUNDS
CATEGORY 23

68N12280 07 01
FIRE PREVENTION
MISSILE SILOS
OXYGEN
CATEGORY 11
HAZARDS
NONFLAMMABLE MATERIALS
SAFETY DEVICES

68N12312 07 05
CAPACITORS
INSULATORS
SEMICONDUCTING FILMS
CATEGORY 9
DETECTORS
METAL OXIDE SEMICONDUCTORS
THIN FILMS

68N15206 13 09
AIRCRAFT SAFETY
DISPLAY DEVICES
FIRE PREVENTION
INTEGRATED CIRCUITS
MICROELECTRONICS
ULTRAVIOLET RADIATION
CATEGORY 8
COMPUTER DESIGN
FAILURE
INFRARED DETECTORS
LOGIC CIRCUITS
TEMPERATURE MEASURING INSTRUMENTS
WARNING SYSTEMS

68N15670 06 01
ACCIDENT INVESTIGATION
CABIN ATMOSPHERES
OXYGEN BREATHING
APOLLO SPACECRAFT
FIRES
CATEGORY 11

68N16903 11 05
AIR
GAS MIXTURES
IGNITION
IGNITION TEMPERATURE
SPACECRAFT CONTAMINATION
CATEGORY 14
ALTITUDE
HYDROGEN
IGNITION LIMITS
SPACECRAFT CABIN ATMOSPHERES
TEMPERATURE DISTRIBUTION

68N17367 11 01
CABIN ATMOSPHERES
FIRES
FLIGHT HAZARDS
IGNITION
OXYGEN
CATEGORY 31
EXTRATERRESTRIAL RESOURCES
FLAME PROPOGATION
HELIUM
NITROGEN
STORAGE

68N17380 16 01
EMERGENCY LIFE SUSTAINING SYSTEMS
FIRE PREVENTION
FLAME PROPOGATION
HUMAN FACTORS ENGINEERING
IGNITION TEMPERATURES
SPACE ENVIRONMENT SIMULATION
SPACECRAFT CABIN ATMOSPHERES
SPONTANEOUS COMBUSTION
ENVIRONMENTAL TESTS
FIREPROOFING
HELMETS
HUMAN FACTORS LABORATORIES
MATERIALS TESTS
SPACE SUITS
SPECIFICATIONS
CATEGORY 5

FIGURE A-1

INPUT DATA FOR SAMPLE PROBLEM

```
DOCUMENT        INDEX TERM   DOCUMENT
 NUMBER            COUNT      UTILITY

68N17925             13          02
BURNING RATE                                 CONTAMINANTS
FIREPROOFING                                 FLAMMABILITY
HAZARDS                                      OUTGASSING
PLASTICS                                     SPACECRAFT CABIN ATMOSPHERES
SPACECRAFT CABINS                            SPACECRAFT CONSTRUCTION MATERIALS
SPACECRAFT CONTAMINATION                     TOXICITY
CATEGORY 5

68N18744             28          04
ACCIDENT INVESTIGATION                       ACCIDENT PREVENTION
BURNS (INJURIES)                             CABIN ATMOSPHERES
CONFERENCES                                  CONTROLLED ATMOSPHERES
ELECTRICAL FAULTS                            EMERGENCY LIFE SUSTAINING SYSTEMS
FIRE CONTROL                                 FIRE EXTINGUISHERS
FIREPROOFING                                 FLAMMABILITY
FREON                                        GAS COMPOSITION
GLASS FIBERS                                 HIGH PRESSURE OXYGEN
HUMAN FACTORS ENGINEERING                    NONFLAMMABLE MATERIALS
OXYGEN                                       PRESSURIZED CABINS
PROTECTIVE CLOTHING                          SAFETY DEVICES
SPACE SUITS                                  SPACECRAFT CABIN SIMULATORS
SPONTANEOUS COMBUSTION                       THERMAL INSULATION
PRESSURE CHAMBERS                            CATEGORY 5

68N18745             13          01
ACCIDENT INVESTIGATION                       CHEMICAL ANALYSIS
CONFERENCES                                  FIRES
HIGH PRESSURE OXYGEN                         HUMAN PATHOLOGY
PRESSURE CHAMBERS                            RESIDUES
SPACECRAFT CABIN SIMULATORS                  SPONTANEOUS COMBUSTION
ELECTRICAL FAULTS                            FLAMMABILITY
CATEGORY 5

68N18746             14          03
CABIN ATMOSPHERES                            CONFERENCES
EMERGENCY LIFE SUSTAINING SYSTEMS            FIRE CONTROL
FIRE EXTINGUISHERS                           FIREPROOFING
HIGH PRESSURE OXYGEN                         HUMAN FACTORS ENGINEERING
NONFLAMMABLE MATERIALS                       PROTECTIVE CLOTHING
SAFETY DEVICES                               SPACECRAFT CABIN SIMULATORS
SURVIVAL                                     CATEGORY 5

68N18747             12          03
ACCIDENT PREVENTION                          CABIN ATMOSPHERES
CONFERENCES                                  EMERGENCY LIFE SUSTAINING SYSTEMS
FIRE CONTROL                                 FIRE EXTINGUISHERS
HUMAN FACTORS ENGINEERING                    PRESSURIZED CABINS
PROTECTIVE CLOTHING                          SAFETY DEVICES
SPACECRAFT CABIN SIMULATORS                  CATEGORY 5

68N18750             12          01
ACCIDENT PREVENTION                          CABIN ATMOSPHERES
EMERGENCY LIFE SUSTAINING SYSTEMS            FIRE CONTROL
FIRE EXTINGUISHERS                           GAS COMPOSITION
HUMAN FACTORS ENGINEERING                    PROTECTIVE CLOTHING
SAFETY DEVICES                               SPACECRAFT CABIN SIMULATORS
SPONTANEOUS COMBUSTION                       CATEGORY 5
```

FIGURE A-2

INPUT DATA FOR SAMPLE PROBLEM

DOCUMENT NUMBER	INDEX TERM COUNT	DOCUMENT UTILITY
68N18751	14	01

CABIN ATMOSPHERES
EMERGENCY LIFE SUSTAINING SYSTEMS
FIREPROOFING
HUMAN FACTORS ENGINEERING
NONFLAMMABLE MATERIALS
SAFETY DEVICES
SPONTANEOUS COMBUSTION
CONFERENCES
FIRE CONTROL
GAS COMPOSITION
MATERIALS TESTS
PROTECTIVE CLOTHING
SPACECRAFT CABIN SIMULATORS
CATEGORY 5

68N20005	12	01

ENVIRONMENT SIMULATION
FLAME PROPAGATION
FLAMMABLE GASES
HIGH PRESSURE OXYGEN
IGNITION
PROTECTIVE CLOTHING
FIRE PREVENTION
FLAMMABILITY
FLASH POINT
HUMAN FACTORS LABORATORIES
PRESSURE DISTRIBUTION
CATEGORY 5

68N20058	12	01

AIRCRAFT HAZARDS
CARBON TETRAFLUORIDE
DIFLUORO COMPOUNDS
FIRE FIGHTING
METHANE
PYROLYSIS
BROMINE COMPOUNDS
CHLORINE FLUORIDES
FIRE EXTINGUISHERS
HALOGEN COMPOUNDS
OXYGEN
CATEGORY 6

68N20870	10	01

COMBUSTION
FIRES
HAZARDS
PROTECTIVE CLOTHING
SPACECRAFT ENVIRONMENTS
EXPLOSIONS
FLAMMABILITY
OXYGEN
SAFETY
CATEGORY 33

68N21752	11	01

FIRE PREVENTION
FLAME PROPAGATION
HUMAN FACTORS LABORATORIES
NONFLAMMABLE MATERIALS
PROTECTIVE CLOTHING
CATEGORY 5
FIREPROOFING
FLAMMABILITY
MICE
OXYGEN
SPACECRAFT CABIN ATMOSPHERES

68N24756	15	01

BIBLIOGRAPHIES
FLAMMABILITY
HEAT TRANSFER
HUMAN TOLERANCES
SPACECRAFT CABIN ATMOSPHERES
SPACECRAFT CONTAMINATION
TOXIC HAZARDS
CATEGORY 5
FIRE EXTINGUISHERS
FLIGHT CREWS
HIGH PRESSURE OXYGEN
LIFE SUPPORT SYSTEMS
SPACECRAFT CONSTRUCTION MATERIALS
STATIC ELECTRICITY
WEIGHTLESSNESS

68N24871	10	01

CONFERENCES
FIRES
IGNITION LIMITS
SPONTANEOUS COMBUSTION
UNITED STATES OF AMERICA
FIRE PREVENTION
GREAT BRITAIN
SPACECRAFT CABIN ATMOSPHERES
THERAPY
CATEGORY 5

68N29668	07	06

AIRCRAFT SAFETY
ELECTROLYTES
ELECTROCHEMICAL CELLS
FIRE PREVENTION

FIGURE A-3

INPUT DATA FOR SAMPLE PROBLEM

DOCUMENT NUMBER	INDEX TERM COUNT	DOCUMENT UTILITY	
TEMPERATURE SENSORS			WARNING SYSTEMS
CATEGORY 14			
68N29947	10	03	
CALIBRATING			CORRECTION
CURRENT AMPLIFIERS			GAS FLOW
INERTIA			SEMICONDUCTOR DEVICES
TEMPERATURE MEASURING INSTRUMENTS			TEMPERATURE SENSORS
TRIODES			CATEGORY 14
68N30134	07	01	
BURNING RATE			FIRE PREVENTION
FLAMMABILITY			IGNITION TEMPERATURE
SPACECRAFT CABIN ATMOSPHERES			SPACECRAFT CONSTRUCTION MATERIALS
CATEGORY 33			
68N34881	11	08	
ATMOSPHERIC COMPOSITION			CLOSED ECOLOGICAL SYSTEMS
ELECTRICAL PROPERTIES			GAS ANALYSIS
ORGANIC COMPOUNDS			POLYMERIC FILMS
SEMICONDUCTING FILMS			SPACECRAFT CABIN ATMOSPHERES
SPACECRAFT CONTAMINATION			THIN FILMS
CATEGORY 5			
68N36272	07	01	
AIRCRAFT FUEL SYSTEMS			CONFERENCES
EXPLOSIONS			FIRE PREVENTION
IGNITION			POLYURETHANE FOAM
CATEGORY 2			
68N36274	12	01	
AIRCRAFT FUEL SYSTEMS			CARBON DIOXIDE
COMMERCIAL AIRCRAFT			CONFERENCES
ELECTRIC DISCHARGES			FIRE PREVENTION
FUEL TANKS			LIGHTNING
LIQUID NITROGEN			SAFETY DEVICES
VENTS			CATEGORY 2
68N36275	08	01	
AIRCRAFT FUEL SYSTEMS			AIRCRAFT HAZARDS
AIRCRAFT INDUSTRY			CONFERENCES
FIRE PREVENTION			JET AIRCRAFT
SAFETY DEVICES			CATEGORY 2

FIGURE A-4

INPUT DATA FOR SAMPLE PROBLEM

DOCUMENT DATA

NO. OF DOCUMENTS PROCESSED=28

CATEGORY THRESHOLD= 3
(DOCUMENTS WITH WEIGHTS GREATER THAN OR EQUAL TO THRESHOLD ARE IN CATEGORY 1)

PROGRAM DOC. NO.	ACTUAL DOC. NO.	DOCUMENT WEIGHT	DOCUMENT CATEGORY	NO. OF TERMS	NEW TERMS
1	68N10674	7	1	10	10
2	68N12280	1	0	7	7
3	68N12312	5	1	7	7
4	68N15206	9	1	13	12
5	68N15620	1	0	6	5
6	68N16903	5	1	11	10
7	68N17367	1	0	11	7
8	68N17380	1	0	16	13
9	68N17925	2	0	13	7
10	68N18744	4	1	28	16
11	68N18745	1	0	13	2
12	68N18746	3	1	14	1
13	68N18747	3	1	12	0
14	68N18750	1	0	12	0
15	68N18751	1	0	14	0
16	68N20005	-1	0	12	5
17	68N20058	1	0	12	10
18	68N20870	1	0	10	5
19	68N21752	1	0	11	1
20	68N24756	1	0	15	7
21	68N24871	1	0	10	3
22	68N29668	6	1	7	3
23	68N29947	3	1	10	7
24	68N30134	1	0	7	0
25	68N34881	8	1	11	5
26	68N36272	1	0	7	3
27	68N36274	1	0	12	7
28	68N36275	1	0	8	2

FIGURE A-5

DOCUMENT DATA SUMMARY FOR SAMPLE PROBLEM

INDEX TERM DATA
ALPHABETICAL SORT

NO. OF TERMS DISCOVERED=155

SOURCE ENTROPY H(X)= 0.940

PROGRAM TERM NO.	INDEX TERM	INFORMATION STATISTIC
37	ACCIDENT INVESTIGATION	0.00022
79	ACCIDENT PREVENTION	0.03441
25	AIRCRAFT SAFETY	0.11340
103	AIRCRAFT HAZARDS	0.04771
144	AIRCRAFT FUEL SYSTEMS	0.07337
154	AIRCRAFT INDUSTRY	0.02329
42	AIR	0.05479
43	ALTITUDE	0.05479
38	APOLLO SPACECRAFT	0.02329
139	ATMOSPHERIC COMPOSITION	0.05479
1	BIBLIOGRAPHIES	0.00474
104	BROMINE COMPOUNDS	0.02329
72	BURNING RATE	0.04771
80	BURNS (INJURIES)	0.05479
39	CABIN ATMOSPHERES	0.00526
132	CALIBRATING	0.05479
18	CAPACITORS	0.05479
105	CARBON TETRAFLUORIDE	0.02329
147	CARBON DIOXIDE	0.02329
10	CATEGORY 23	0.05479
17	CATEGORY 11	0.04771
24	CATEGORY 9	0.05479
36	CATEGORY 8	0.05479
51	CATEGORY 14	0.17649
58	CATEGORY 31	0.02329
71	CATEGORY 5	0.00669
112	CATEGORY 6	0.02329
117	CATEGORY 33	0.04771
146	CATEGORY 2	0.07337
2	CHEMICAL ANALYSIS	0.00474
106	CHLORINE FLUORIDES	0.02329
140	CLOSED ECOLOGICAL SYSTEM	0.05479
113	COMBUSTION	0.02329
148	COMMERCIAL AIRCRAFT	0.02329
26	COMPUTER DESIGN	0.05479
81	CONFERENCES	0.00085
3	CONTAMINANTS	0.00474
82	CONTROLLED ATMOSPHERES	0.05477
133	CORRECTION	0.05479
134	CURRENT AMPLIFIERS	0.05479
19	DETECTORS	0.05479
107	DIFLUORO COMPOUNDS	0.02329
27	DISPLAY DEVICES	0.05479
83	ELECTRICAL FAULTS	0.00474
129	ELECTROCHEMICAL CELLS	0.05479
130	ELECTROLYTES	0.05479
141	ELECTRICAL PROPERTIES	0.05479
149	ELECTRIC DISCHARGES	0.02329
59	EMERGENCY LIFE SUSTAININ	0.01698
60	ENVIRONMENTAL TESTS	0.02329

FIGURE A-6

ALPHABETICAL LISTING OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

INDEX TERM NUMBER	INDEX TERM	INFORMATION STATISTIC
98	ENVIRONMENT SIMULATION	0.02329
114	EXPLOSIONS	0.04771
52	EXTRATERRESTRIAL RESOURC	0.02329
28	FAILURE	0.05479
61	FIREPROOFING	0.00049
11	FIRE PREVENTION	0.06593
40	FIRES	0.12897
84	FIRE CONTROL	0.03867
85	FIRE EXTINGUISHERS	0.01698
108	FIRE FIGHTING	0.02329
53	FLAME PROPOGATION	0.04771
99	FLAME PROPAGATION	0.04771
73	FLAMMABILITY	0.07586
100	FLAMMABLE GASES	0.02329
101	FLASH POINT	0.02329
54	FLIGHT HAZARDS	0.02329
119	FLIGHT CREWS	0.02329
86	FREON	0.05479
150	FUEL TANKS	0.02329
87	GAS COMPOSITION	0.00022
142	GAS ANALYSIS	0.05479
135	GAS FLOW	0.05479
44	GAS MIXTURES	0.05479
88	GLASS FIBERS	0.05479
126	GREAT BRITAIN	0.02329
109	HALOGEN COMPOUNDS	0.02329
12	HAZARDS	0.07337
120	HEAT TRANSFER	0.02329
55	HELIUM	0.02329
62	HELMETS	0.02329
89	HIGH PRESSURE OXYGEN	0.00124
63	HUMAN FACTORS ENGINEERIN	0.01698
64	HUMAN FACTORS LABORATORI	0.07337
95	HUMAN PATHOLOGY	0.02329
121	HUMAN TOLERANCES	0.02329
45	HYDROGEN	0.05479
46	IGNITION	0.00630
47	IGNITION LIMITS	0.00474
48	IGNITION TEMPERATURE	0.00474
65	IGNITION TEMPERATURES	0.02329
136	INERTIA	0.05479
29	INFRARED DETECTORS	0.05479
4	INORGANIC COMPOUNDS	0.05479
20	INSULATORS	0.05479
30	INTEGRATED CIRCUITS	0.05479
155	JET AIRCRAFT	0.02329
122	LIFE SUPPORT SYSTEMS	0.02329
151	LIGHTNING	0.02329
152	LIQUID NITROGEN	0.02329
31	LOGIC CIRCUITS	0.05479
66	MATERIALS TESTS	0.04771
21	METAL OXIDE SEMICONDUCTO	0.05479
110	METHANE	0.02329
118	MICE	0.02329
5	MICROWAVE SPECTRA	0.05479
32	MICROELECTRONICS	0.05479
13	MISSILE SILOS	0.02329
6	MOLECULAR SPECTROSCOPY	0.05479
7	MOLECULAR STRUCTURE	0.05479
56	NITROGEN	0.02329

FIGURE A-7

ALPHABETICAL LISTING OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

INDEX TERM NUMBER	INDEX TERM	INFORMATION STATISTIC
14	NONFLAMMABLE MATERIALS	0.00124
8	ORGANIC COMPOUNDS	0.11340
74	OUTGASSING	0.02329
15	OXYGEN	0.03412
41	OXYGEN BREATHING	0.02329
75	PLASTICS	0.02329
143	POLYMERIC FILMS	0.05479
145	POLYURETHANE FOAM	0.02329
90	PRESSURIZED CABINS	0.11340
94	PRESSURE CHAMBERS	0.00474
102	PRESSURE DISTRIBUTION	0.02329
91	PROTECTIVE CLOTHING	0.00040
111	PYROLYSIS	0.02329
96	RESIDUES	0.02329
16	SAFETY DEVICES	0.00040
115	SAFETY	0.02329
22	SEMICONDUCTING FILMS	0.11340
137	SEMICONDUCTOR DEVICES	0.05479
9	SPACECRAFT CABIN ATMOSPH	0.00085
49	SPACECRAFT CONTAMINATION	0.01032
67	SPACE ENVIRONMENT SIMULA	0.02329
68	SPACE SUITS	0.00474
76	SPACECRAFT CABINS	0.02329
77	SPACECRAFT CONSTRUCTION	0.07337
92	SPACECRAFT CABIN SIMULAT	0.01698
116	SPACECRAFT ENVIRONMENTS	0.02329
69	SPECIFICATIONS	0.02329
70	SPONTANEOUS COMBUSTION	0.03412
123	STATIC ELECTRICITY	0.02329
57	STORAGE	0.02329
97	SURVIVAL	0.05479
33	TEMPERATURE MEASURING IN	0.11340
50	TEMPERATURE DISTRIBUTION	0.05479
131	TEMPERATURE SENSORS	0.11340
127	THERAPY	0.02329
93	THERMAL INSULATION	0.05479
23	THIN FILMS	0.11340
78	TOXICITY	0.02329
124	TOXIC HAZARDS	0.02329
138	TRIODES	0.05479
34	ULTRAVIOLET RADIATION	0.05479
128	UNITED STATES OF AMERICA	0.02329
153	VENTS	0.02329
35	WARNING SYSTEMS	0.11340
125	WEIGHTLESSNESS	0.02329

FIGURE A-8

ALPHABETICAL LISTING OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

INDEX TERM DATA
INFO. STAT. SORT

NO. OF TERMS DISCOVERED=155

SOURCE ENTROPY H(X)= 0.940

PROGRAM TERM NO.	INDEX TERM	INFORMATION STATISTIC
51	CATEGORY 14	0.17649
40	FIRES	0.12897
131	TEMPERATURE SENSORS	0.11340
90	PRESSURIZED CABINS	0.11340
35	WARNING SYSTEMS	0.11340
33	TEMPERATURE MEASURING IN	0.11340
25	AIRCRAFT SAFETY	0.11340
23	THIN FILMS	0.11340
22	SEMICONDUCTING FILMS	0.11340
8	ORGANIC COMPOUNDS	0.11340
73	FLAMMABILITY	0.07586
146	CATEGORY 2	0.07337
144	AIRCRAFT FUEL SYSTEMS	0.07337
77	SPACECRAFT CONSTRUCTION	0.07337
64	HUMAN FACTORS LABORATORI	0.07337
12	HAZARDS	0.07337
11	FIRE PREVENTION	0.06593
143	POLYMERIC FILMS	0.05479
142	GAS ANALYSIS	0.05479
141	ELECTRICAL PROPERTIES	0.05479
140	CLOSED ECOLOGICAL SYSTEM	0.05479
139	ATMOSPHERIC COMPOSITION	0.05479
138	TRIODES	0.05479
137	SEMICONDUCTOR DEVICES	0.05479
136	INERTIA	0.05479
135	GAS FLOW	0.05479
134	CURRENT AMPLIFIERS	0.05479
133	CORRECTION	0.05479
132	CALIBRATING	0.05479
130	ELECTROLYTES	0.05479
129	ELECTROCHEMICAL CELLS	0.05479
97	SURVIVAL	0.05479
93	THERMAL INSULATION	0.05479
88	GLASS FIBERS	0.05479
86	FREON	0.05479
82	CONTROLLED ATMOSPHERES	0.05479
80	BURNS (INJURIES)	0.05479
50	TEMPERATURE DISTRIBUTION	0.05479
45	HYDROGEN	0.05479
44	GAS MIXTURES	0.05479
43	ALTITUDE	0.05479
42	AIR	0.05479
36	CATEGORY 8	0.05479
34	ULTRAVIOLET RADIATION	0.05479
32	MICROELECTRONICS	0.05479
31	LOGIC CIRCUITS	0.05479
30	INTEGRATED CIRCUITS	0.05479
29	INFRARED DETECTORS	0.05479
28	FAILURE	0.05479
27	DISPLAY DEVICES	0.05479

FIGURE A-9

INFORMATION STATISTIC SORT OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

INDEX TERM NUMBER	INDEX TERM	INFORMATION STATISTIC
26	COMPUTER DESIGN	0.05479
24	CATEGORY 9	0.05479
21	METAL OXIDE SEMICONDUCTO	0.05479
20	INSULATORS	0.05479
19	DETECTORS	0.05479
18	CAPACITORS	0.05479
10	CATEGORY 23	0.05479
7	MOLECULAR STRUCTURE	0.05479
6	MOLECULAR SPECTROSCOPY	0.05479
5	MICROWAVE SPECTRA	0.05479
4	INORGANIC COMPOUNDS	0.05479
117	CATEGORY 33	0.04771
114	EXPLOSIONS	0.04771
103	AIRCRAFT HAZARDS	0.04771
99	FLAME PROPAGATION	0.04771
72	BURNING RATE	0.04771
66	MATERIALS TESTS	0.04771
53	FLAME PROPOGATION	0.04771
17	CATEGORY 11	0.04771
84	FIRE CONTROL	0.03867
79	ACCIDENT PREVENTION	0.03441
70	SPONTANEOUS COMBUSTION	0.03412
15	OXYGEN	0.03412
155	JET AIRCRAFT	0.02329
154	AIRCRAFT INDUSTRY	0.02329
153	VENTS	0.02329
152	LIQUID NITROGEN	0.02329
151	LIGHTNING	0.02329
150	FUEL TANKS	0.02329
149	ELECTRIC DISCHARGES	0.02329
148	COMMERCIAL AIRCRAFT	0.02329
147	CARBON DIOXIDE	0.02329
145	POLYURETHANE FOAM	0.02329
128	UNITED STATES OF AMERICA	0.02329
127	THERAPY	0.02329
126	GREAT BRITAIN	0.02329
125	WEIGHTLESSNESS	0.02329
124	TOXIC HAZARDS	0.02329
123	STATIC ELECTRICITY	0.02329
122	LIFE SUPPORT SYSTEMS	0.02329
121	HUMAN TOLERANCES	0.02329
120	HEAT TRANSFER	0.02329
119	FLIGHT CREWS	0.02329
118	MICE	0.02329
116	SPACECRAFT ENVIRONMENTS	0.02329
115	SAFETY	0.02329
113	COMBUSTION	0.02329
112	CATEGORY 6	0.02329
111	PYROLYSIS	0.02329
110	METHANE	0.02329
109	HALOGEN COMPOUNDS	0.02329
108	FIRE FIGHTING	0.02329
107	DIFLUORO COMPOUNDS	0.02329
106	CHLORINE FLUORIDES	0.02329
105	CARBON TETRAFLUORIDE	0.02329
104	BROMINE COMPOUNDS	0.02329
102	PRESSURE DISTRIBUTION	0.02329
101	FLASH POINT	0.02329
100	FLAMMABLE GASES	0.02329
98	ENVIRONMENT SIMULATION	0.02329

FIGURE A-10

INFORMATION STATISTIC SORT OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

INDEX TERM NUMBER	INDEX TERM	INFORMATION STATISTIC
96	RESIDUES	0.02329
95	HUMAN PATHOLOGY	0.02329
78	TOXICITY	0.02329
76	SPACECRAFT CABINS	0.02329
75	PLASTICS	0.02329
74	OUTGASSING	0.02329
69	SPECIFICATIONS	0.02329
67	SPACE ENVIRONMENT SIMULA	0.02329
65	IGNITION TEMPERATURES	0.02329
62	HELMETS	0.02329
60	ENVIRONMENTAL TESTS	0.02329
58	CATEGORY 31	0.02329
57	STORAGE	0.02329
56	NITROGEN	0.02329
55	HELIUM	0.02329
54	FLIGHT HAZARDS	0.02329
52	EXTRATERRESTRIAL RESOURC	0.02329
41	OXYGEN BREATHING	0.02329
38	APOLLO SPACECRAFT	0.02329
13	MISSILE SILOS	0.02329
92	SPACECRAFT CABIN SIMULAT	0.01698
85	FIRE EXTINGUISHERS	0.01698
63	HUMAN FACTORS ENGINEERIN	0.01698
59	EMERGENCY LIFE SUSTAININ	0.01698
49	SPACECRAFT CONTAMINATION	0.01032
71	CATEGORY 5	0.00669
46	IGNITION	0.00630
39	CABIN ATMOSPHERES	0.00526
94	PRESSURE CHAMBERS	0.00474
83	ELECTRICAL FAULTS	0.00474
68	SPACE SUITS	0.00474
48	IGNITION TEMPERATURE	0.00474
47	IGNITION LIMITS	0.00474
3	CONTAMINANTS	0.00474
2	CHEMICAL ANALYSIS	0.00474
1	BIBLIOGRAPHIES	0.00474
89	HIGH PRESSURE OXYGEN	0.00124
14	NONFLAMMABLE MATERIALS	0.00124
81	CONFERENCES	0.00085
9	SPACECRAFT CABIN ATMOSPH	0.00085
61	FIREPROOFING	0.00049
91	PROTECTIVE CLOTHING	0.00040
16	SAFETY DEVICES	0.00040
87	GAS COMPOSITION	0.00022
37	ACCIDENT INVESTIGATION	0.00022

FIGURE A-11

INFORMATION STATISTIC SORT OF INDEX TERMS IN SAMPLE PROBLEM TRAINING SET

DOCUMENT-TERM MATRIX

	DOCUMENT NUMBER																												
UTILITY	7	1	5	9	1	5	1	1	2	4	1	3	3	1	1	1	1	1	1	1	1	6	3	1	8	1	1	1	
CATEGORY	1	0	1	1	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	
INDEX TERM	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	INFORMATION STATISTIC
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.00474
2	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00474
3	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00474
4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.11340
9	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0.00085
10	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
11	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	1	0.06593
12	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0.07337
13	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
14	0	1	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0.00124
15	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0.03412
16	0	1	0	0	0	0	0	0	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0.00040
17	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.04771
18	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
19	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
20	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
21	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
22	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.11340
23	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.11340
24	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
25	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0.11340
26	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
27	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
28	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
29	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
30	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
31	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
32	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
33	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.11340
34	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
35	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0.11340
36	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
37	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00022
38	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
39	0	0	0	0	1	0	1	0	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00526
40	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0.12897
41	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
42	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
43	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
44	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
45	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
46	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0.00630
47	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	0.00474
48	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0.00474
49	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0.01032
50	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
51	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0.17649
52	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
53	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.04771

FIGURE A-12

DOCUMENT-TERM MATRIX FOR SAMPLE PROBLEM

INDEX TERM	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	INFORMATION STATISTIC
54	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
55	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
56	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
57	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
58	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
59	0	0	0	0	0	0	0	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01698
60	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
61	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0.00049
62	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
63	0	0	0	0	0	0	0	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01698
64	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0.07337
65	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
66	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.04771
67	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02379
68	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00474
69	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
70	0	0	0	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0.03412
71	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	1	1	1	0	0	0	1	0	0	0	0.00669
72	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	0.04771
73	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	0	1	1	1	0	0	0	1	0	0	0	0	0.07586
74	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
75	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
76	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
77	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0.07337
78	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02329
79	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.03441
80	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
81	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	0.00085
82	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
83	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00474
84	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.03667
85	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0.01698
86	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
87	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00022
88	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
89	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0.00124
90	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.11340
91	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0.00040
92	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01698
93	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05479
94	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00474
95	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	0	0	0	0.02329
96	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	0	0	0	0.02329
97	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	0	0	0.05479
98	0	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	0.02329
99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0.04771
100	0	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	0.02329
101	0	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	0.02329
102	0	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	0.02329
103	0	0	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	1	0.04771
104	0	0	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	0.02329
105	0	0	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	0.02329
106	0	0	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	0.02329
107	0	0	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	0.02329
108	0	0	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	0.02329
109	0	0	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	0.02329
110	0	0	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	0.02329
111	0	0	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	0.07329
112	0	0	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	0.02329
113	0	0	0	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	0.02329

FIGURE A-13

DOCUMENT-TERM MATRIX FOR SAMPLE PROBLEM

INDEX TERM	DOCUMENT NUMBER 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	INFORMATION STATISTIC
114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0.04771
115	0	0	0	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	0.02329
116	0	0	0	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	0.02329
117	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0.04771
118	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0.02329
119	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
121	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
122	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
123	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
124	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.02329
126	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0.02329
127	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0.02329
128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0.02329
129	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0.05479
130	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0.05479
131	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0.11340
132	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
133	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
135	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
136	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
137	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
138	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.05479
139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.05479
140	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.05479
141	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.05479
142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.05479
143	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.05479
144	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0.07337
145	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0.02329
146	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0.07337
147	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
148	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
149	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
151	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.02329
154	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.02329
155	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.02329

FIGURE A-14

DOCUMENT-TERM MATRIX FOR SAMPLE PROBLEM

COLUMN NUMBER	MATRIX COLUMN ELEMENTS																		COST COEFFICIENTS
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00									
2	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	1.00	1.00	0.	0.	0.	0.	0.									
3	0.	0.	0.	0.	1.00	0.	1.00	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	1.00	0.
	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.									
4	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	1.00	1.00	0.	0.	0.	0.	0.									
5	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	1.00	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
6	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.									
7	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.									
8	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.									
9	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.									
10	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.									
11	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.									
12	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	0.
	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00									
13	-0.	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-1.00	-1.00	-0.	-0.	-0.	-0.	-0.									
14	-0.	-0.	-0.	-0.	-1.00	-0.	-1.00	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-1.00	0.
	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.									
15	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-1.00	-1.00	-0.	-0.	-0.	-0.	-0.									
16	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.									
17	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.									
18	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.									
19	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.									
20	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-0.									
21	-0.	-0.	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-0.									
22	-1.00	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	-0.	0.
	-0.	-0.	-0.	-0.	-0.	-0.	-1.00	-0.	-0.	-0.									
23	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
24	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
25	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
26	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
27	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
28	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000

FIGURE A-15

LP STRUCTURAL MATRIX

COLUMN NUMBER	MATRIX COLUMN ELEMENTS																		COST COEFFICIENTS
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
29	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
30	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
31	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
32	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
33	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
34	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
35	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
36	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
37	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
38	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
39	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
40	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
41	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.									
42	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	1.00	0.	0.	0.	0.	0.	0.	0.	0.									
43	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	1.00	0.	0.	0.	0.	0.	0.	0.									
44	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	1.00	0.	0.	0.	0.	0.	0.									
45	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	1.00	0.	0.	0.	0.	0.									
46	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	1.00	0.	0.	0.	0.									
47	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	1.00	0.	0.	0.									
48	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	1.00	0.	0.									
49	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	1.00	0.									
50	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.00									
51	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
52	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
53	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
54	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
55	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
56	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
57	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
58	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000

FIGURE A-16

LP STRUCTURAL MATRIX

COLUMN NUMBER	MATRIX COLUMN ELEMENTS																		COST COEFFICIENTS
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
59	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
60	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
61	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
62	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
63	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
64	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
65	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
66	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
67	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
68	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.									
69	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	-1.00	0.	0.	0.	0.	0.	0.	0.	0.	0.									
70	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	-1.00	0.	0.	0.	0.	0.	0.	0.	0.									
71	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	-1.00	0.	0.	0.	0.	0.	0.	0.									
72	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	-1.00	0.	0.	0.	0.	0.	0.									
73	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	-1.00	0.	0.	0.	0.	0.									
74	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	-1.00	0.	0.	0.	0.									
75	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	-1.00	0.	0.	0.									
76	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	-1.00	0.	0.									
77	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	-1.00	0.									
78	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.0000
	0.	0.	0.	0.	0.	0.	0.	0.	0.	-1.00									

RIGHT HAND SIDE

7.00	1.00	5.00	9.00	1.00	5.00	1.00	1.00	2.00	4.00	1.00	3.00	3.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	6.00	3.00	1.00	8.00	1.00	1.00	1.00								

FIGURE A-17

LP STRUCTURAL MATRIX

BASIC SOLUTION SUMMARY
(PROGRAM TERM NO.=-1 FOR SLACK,0 FOR CONSTANT,-2 FOR ARTIF.)
(BASIC VAR. NO.=0 FOR ARTIF. VAR.)

BASIC VAR NO.	PROGRAM TERM NO.	INDEX TERM		TERM WEIGHT	BASIC VARIABLE TYPE	INF. STAT.
23	-1	68N10674	SLACK	3.00000	REG.	
1	0	CONST.		1.00000	REG.	0.
9	23	THIN FILMS		4.00000	REG.	0.11340
15	131	TEMPERATURE SENSORS		-4.50000	REG.	0.11340
24	-1	68N12280	SLACK	0.	REG.	
2	51	CATEGORY 14		4.00000	REG.	0.17649
55	-1	68N15620	SLACK	-0.	REG.	
3	40	FIRES		0.	REG.	0.12897
31	-1	68N17925	SLACK	1.00000	REG.	
32	-1	68N18744	SLACK	1.00000	REG.	
57	-1	68N17367	SLACK	-0.	REG.	
34	-1	68N18746	SLACK	2.00000	REG.	
5	90	PRESSURIZED CABINS		2.00000	REG.	0.11340
64	-1	68N18750	SLACK	0.	REG.	
65	-1	68N18751	SLACK	0.	REG.	
66	-1	68N20005	SLACK	0.	REG.	
67	-1	68N20058	SLACK	0.	REG.	
40	-1	68N20870	SLACK	0.	REG.	
69	-1	68N21752	SLACK	0.	REG.	
70	-1	68N24756	SLACK	0.	REG.	
43	-1	68N24871	SLACK	-0.	REG.	
6	35	WARNING SYSTEMS		5.50000	REG.	0.11340
7	33	TEMPERATURE MEASURING IN		2.50000	REG.	0.11340
74	-1	68N30134	SLACK	0.	REG.	
11	8	ORGANIC COMPOUNDS		3.00000	REG.	0.11340
48	-1	68N36272	SLACK	0.	REG.	
49	-1	68N36274	SLACK	0.	REG.	
50	-1	68N36275	SLACK	0.	REG.	

OPTIMAL VALUE OF OBJECTIVE FUNCTION= 7.00000

FIGURE A-18

LP SOLUTION SUMMARY

DEPENDENT VARIABLE	MEASURED VALUE	PREDICTED VALUE	RESIDUAL
1	7.00000	4.00000	3.00000
2	1.00000	1.00000	0.
3	5.00000	5.00000	0.
4	9.00000	9.00000	0.
5	1.00000	1.00000	0.
6	5.00000	5.00000	0.
7	1.00000	1.00000	0.
8	1.00000	1.00000	0.
9	2.00000	1.00000	1.00000
10	4.00000	3.00000	1.00000
11	1.00000	1.00000	0.
12	3.00000	1.00000	2.00000
13	3.00000	3.00000	0.
14	1.00000	1.00000	0.
15	1.00000	1.00000	0.
16	1.00000	1.00000	0.
17	1.00000	1.00000	0.
18	1.00000	1.00000	0.
19	1.00000	1.00000	0.
20	1.00000	1.00000	0.
21	1.00000	1.00000	0.
22	6.00000	6.00000	0.
23	3.00000	3.00000	0.
24	1.00000	1.00000	0.
25	8.00000	8.00000	0.
26	1.00000	1.00000	0.
27	1.00000	1.00000	0.
28	1.00000	1.00000	0.

LENGTH OF RESIDUAL VECTOR (L1 SENSE)= 7.00000

FIGURE A-19

COMPUTATION OF RESIDUALS FOR FITTED MODEL

A. ORIGINAL COEFFICIENTS

RIGHT HAND SIDE= 20

VAR NO.	INDEX TERM	INTEGER COEFFICIENT	COMPLEMENT
1	THIN FILMS	40	0
2	TEMPERATURE SENSORS	-45	0
3	CATEGORY 14	40	0
4	PRESSURIZED CABINS	20	0
5	WARNING SYSTEMS	55	0
6	TEMPERATURE MEASURING IN	25	0
7	ORGANIC COMPOUNDS	30	0

B. CANONICAL FORM COEFFICIENTS

RIGHT HAND SIDE= 65

J	COEF(J)	COMPL(J)	ORDER(J)
1	55	0	5
2	45	1	2
3	40	0	3
4	40	0	1
5	30	0	7
6	25	0	6
7	20	0	4

FIGURE A-20

INITIAL AND CANONICAL FORM COEFFICIENTS FOR THE PSEUDO-BOOLEAN INEQUALITY

A. BASIC SOLUTIONS OF CANONICAL INEQUALITY

SOLUTION NUMBER	VARIABLE 1	2	3	4	5	6	7
1	0	0	0	0	1	1	1
2	0	0	0	1	1	0	0
3	0	0	0	1	0	1	0
4	0	0	1	1	0	0	0
5	0	0	1	0	1	0	0
6	0	0	1	0	0	1	0
7	0	1	1	0	0	0	0
8	0	1	0	1	0	0	0
9	0	1	0	0	1	0	0
10	0	1	0	0	0	1	0
11	0	1	0	0	0	0	1
12	1	1	0	0	0	0	0
13	1	0	1	0	0	0	0
14	1	0	0	1	0	0	0
15	1	0	0	0	1	0	0
16	1	0	0	0	0	1	0
17	1	0	0	0	0	0	1

B SOLUTION FAMILIES OF ORIGINAL INEQUALITY

SOLUTION NUMBER	VARIABLE 1	2	3	4	5	6	7	MIN	BASE	MAX	RHS	SIZE
1	0	1	0	1	0	1	1	30	30	30	65	4
2	1	1	0	2	0	2	1	25	25	70	65	3
3	1	1	0	2	0	1	0	20	20	40	65	3
4	1	1	1	2	0	2	2	35	35	110	65	3
5	0	1	1	2	0	2	1	25	25	70	65	3
6	0	1	1	2	0	1	0	20	20	40	65	3
7	2	0	1	2	0	2	2	40	40	155	65	1
8	1	0	0	2	0	2	2	40	40	115	65	1
9	0	0	0	2	0	2	1	30	30	75	65	1
10	0	0	0	2	0	1	0	25	25	45	65	1
11	0	0	0	1	0	0	0	20	20	20	65	1
12	2	0	2	2	1	2	2	55	55	210	65	1
13	2	1	1	2	1	2	2	50	50	165	65	3
14	1	1	0	2	1	2	2	50	50	125	65	3
15	0	1	0	2	1	2	1	40	40	85	65	3
16	0	1	0	2	1	1	0	35	35	55	65	3
17	0	1	0	1	1	0	0	30	30	30	65	3

FIGURE A-21

SOLUTIONS TO THE PSEUDO-BOOLEAN INEQUALITY

BIBLIOGRAPHY

1. The NASA Scientific and Technical Information System: Its Scope and Coverage, NASA Brochure, (Washington: NASA Scientific and Technical Information Division; March, 1970).

2. W.T. Brandhorst and P. F. Eckert, Guide to the Processing, Storage, and Retrieval of Bibliographic Information at the NASA Scientific and Technical Information Facility, NASA Contractor Report CR-62033, N66-34085, (Washington: U.S. Government Printing Office, 1966).

3. IBM Corp., Implementation, Test, and Evaluation of a Selective Dissemination System for NASA Scientific and Technical Information: Final Report, NASA Contractor Report CR 62020, N67-11241, (Washington: U.S. Government Printing Office, 1966).

4. Introducing NASA's RECON, NASA Brochure, (Washington: NASA Scientific and Technical Information Division, Office of Technology Utilization, 1969).

5. G. Salton, Automatic Information Organization and Retrieval, (New York: McGraw Hill, 1968).

6. D.J. Hillman, "Negotiation of Inquiries in an on-Line Retrieval System," Information Storage and Retrieval, Vol. 4, No. 2 (June, 1968), pp. 219-238.

7. C.T. Meadow, The Analysis of Information Systems; an Introduction to Information Retrieval, (New York: John Wiley, 1967).

8. D.T. Komoto, "Wesrac System," Datamation, Vol. 16, No. 9 (Aug., 1970), pp. 43-47.

9. Salton, op.cit, pp. 243-252.

10. M.E. Maron and J.L. Kuhns, "On Relevance, Probabilistic Indexing and Information Retrieval," J. Assoc. Comp. Mach., Vol. 7, No. 3 (July, 1960), pp. 216-244.

11. J.H. Williams, "A Discriminant Method for Automatically Classifying Documents," Proc. Fall Joint Computer Conf., Vol. 24 (1963), pp. 161-166.

12. Hillman, loc. cit.

13. R.T. Brandhorst, "Simulation of Boolean Logic Constraints through the Use of Term Weights," American Documentation, Vol. 17, No. 3 (1966), pp. 145-146.

14. H.P. Eker, "Solution of Boolean Equations through the Use of Term Weights to the Base Two," American Documentation, Vol. 18, No. 1 (1967), p. 47.

15. P.M. Lewis, "The Characteristic Selection Problem in Recognition Systems," IRE Trans. Infor. Theory, Vol. IT-8, No. 2 (Feb., 1962), pp. 171-178.

16. C. Maltz, "A Measure of the Significance of Pattern Features for Use as an Aid in the Design of Recognition Systems" (unpublished M.S. Thesis, Dept. of Engineering, University of California at Los Angeles, 1962), pp. 98-102.

17. S. Watanabe, Knowing and Guessing, A Quantitative Study of Inference and Information (New York: John Wiley, 1969), pp. 1-27.

18. I. Barrodale and A. Young, "Algorithms for Best L_1 and L_∞ Linear Approximations on a Discrete Set," Numerische Mathematik, Vol. 8, (1966), pp. 295-306.

19. I. Barrodale, "Approximation in the L_1 and L_∞ Norms by Linear Programming" (unpublished Ph.D. dissertation, Dept. of Computational and Statistical Science, University of Liverpool, 1967).

20. P.L. Hammer and S. Rudeanu, Pseudo-Boolean Methods for Bivalent Programming, Lecture Notes in Mathematics, Vol. 23, (Berlin, Heidelberg, New York: Springer-Verlag, 1966), pp. 23-26.

21. P.L. Hammer and S. Rudeanu, Boolean Methods in Operations Research, (New York: Springer-Verlag, 1968), pp. 54-65.

22. P.L. Hammer and S. Rudeanu, "Pseudo-Boolean Programming," Operations Research, Vol. 17, No. 3 (1969), pp. 233-261.

23. D.E. Knuth, The Art of Computer Programming, Vol. 1, Fundamental Algorithms, (Reading, Mass.: Addison-Wesley, 1968), pp. 305-328.

24. D.W. King and E.C. Bryant, Evaluation of Document Retrieval Systems: Literature Perspective, Measurement, Technical Papers, Part II. Measurement. Prepared by Westat Research, Inc. for Office of Science Information Service, National Science Foundation Report No. PB. 182-710, (Springfield, Va.: Clearinghouse for Federal Scientific and Technical Information, 1969).

25. F.W. Lancaster, Information Retrieval Systems: Characteristics, Testing, Evaluation, (New York: John Wiley, 1968).

26. A.R. Meetham, "Communication Theory and the Evaluation of Information Retrieval Systems," Inform. Stor. Ret., Vol. 5, No. 3 (Oct., 1969), pp. 129-134.

27. E.W. Kozdrowicki, "An Adaptive Tree Pruning System: A Language for Programming Heuristic Tree Searches" (unpublished Ph.D. dissertation, Dept. of Electrical Engineering, University of Washington, 1967).

28. N.J. Nilsson, Learning Machines, (New York: McGraw-Hill, 1965).

29. G. Nagy, "State of the Art in Pattern Recognition," Proc. IEEE, Vol. 56, No. 5 (May, 1968), pp. 836-862.

30. J. Ho and A.K. Agrawala, "On Pattern Classification Algorithms - Introduction and Survey," Proc. IEEE, Vol. 56, No. 12 (Dec., 1968), pp. 2101-2114.

31. P.E. Hart, A Brief Survey of Preprocessing for Pattern Recognition, Rome Air Development Center Technical Report RADC-TR-66-819; also AD-647275, N67-24942, (Springfield, Va.: Clearinghouse for Federal Scientific and Technical Information, 1967).

32. P.J. Davis, Interpolation and Approximation, (New York: Blaisdell, 1963), pp. 136-140.

33. J.R. Rice, The Approximation of Functions, Vol. 1, Linear Theory, (Reading, Mass.: Addison-Wesley, 1964), p. 11.

34. Davis, op.cit., pp. 128-134.

35. Rice, op.cit., pp. 4-8.

36. F.A. Graybill, An Introduction to Linear Statistical Models, Vol. 1, (New York: McGraw-Hill, 1961), pp. 117-120.

37. Barrodale and Young, op. cit.

38. Barrodale, op. cit.

39. P. Rabinowitz, "Applications of Linear Programming to Numerical Analysis," SIAM Review, Vol. 10, No. 2 (April, 1968), pp. 121-159.

40. Graybill, op. cit., pp. 110-114.

41. J.R. Rice and J.S. White, "Norms for Smoothing and Estimation," SIAM Review, Vol. 6, No. 3 (1964), pp. 243-265.

42. Davis, op. cit., pp. 140-146.

43. F.W. Smith, "Pattern Classifier Design by Linear Programming," IEEE Trans. on Comp., Vol. C-17, No. 4 (April, 1968), pp. 367-372.

44. R.C. Grinold, "A Note on Pattern Separation," Operations Research, Vol. 18, No. 1 (Jan.-Feb., 1970), pp. 187-190.

45. Y.C. Ho and R.L. Kashyap, "An Algorithm for Linear Inequalities and Its Application," IEEE Trans. Elect. Comp., Vol. EC-14, No. 5 (Oct., 1965), pp. 683-688.

46. O.L. Mangasarian, "Linear and Nonlinear Separation of Patterns by Linear Programming," Operations Research, Vol. 13 (1965), pp. 444-452.

47. M. Taylor, Pattern Separation by Linear Programming, Ballistic Research Labs. Report, Aberdeen-Proving Ground, Maryland; AD-656-928, N67-38194, (Springfield, Va.: Clearinghouse for Federal Scientific and Technical Information, 1965).

48. Graybill, op. cit.

49. N.R. Draper and H. Smith, Applied Regression Analysis, (New York: John Wiley, 1966).

50. R.L. Kashyap and C.C. Blaydon, "Recovery of Functions from Noisy Measurements Taken at Randomly Selected Points and Its Application to Pattern Recognition," Proc. IEEE, Vol. 54, No. 8 (Aug., 1966), pp. 1127-1129.

51. Graybill, op. cit., p. 104 and pp. 223-253.

52. Hammer and Rudeanu, "Pseudo-Boolean Programming," op. cit.

53. *The NASA Scientific and Technical Information System ...*, op. cit.

54. Brandhorst and Eckert, op. cit.

55. IBM Corp., *Implementation,...,* op. cit.

56. *Introducing NASA's RECON*, op. cit.

57. P. Fishburn, *Decision and Value Theory*, (New York: John Wiley, 1964).

58. G. Hadley, *Introduction to Probability and Statistical Decision Theory*, (San Francisco: Holden-Day, 1967), pp. 160-166.

59. P.A.V. Hall, "Pattern Classification as Interpolation in N Dimensions," *The Computer Journal*, Vol. 11, No. 3 (Nov., 1968), pp. 287-292.

60. Hadley, op. cit.

61. Fishburn, op. cit.

62. Hadley, op. cit., pp. 418-491.

63. Ibid., pp. 296-298.

64. A. Feinstein, *Foundations of Information Theory*, (New York: McGraw-Hill, 1958), pp. 11-23.

65. Watanabe, op. cit.

66. Feinstein, op. cit., pp. 4-9.

67. Watanabe, op. cit., pp. 105-114.

68. Ibid.

69. J.C. Hancock, *An Introduction to the Principles of Communication Theory*, (New York: McGraw-Hill, 1961), pp. 168-169.

70. S. Kullback, M. Kupperman, and H.H. Ku, "An Application of Information Theory to the Analysis of Contingency Tables with a Table of $(2n)\ell n(n)$, $n=1(1)10{,}000$," *Journal of Research of the National Bureau of Standards - B. Mathematics and Mathematical Physics.*, Vol. 66B, No. 4 (Oct. - Dec., 1962), pp. 217-243.

71. Ibid.

72. Rice, op. cit., pp. 4-8.

73. Davis, op. cit., pp. 128-134.

74. Barrodale, op. cit., pp. 71-74.

75. Barrodale and Young, op. cit.

76. Rabinowitz, op. cit., pp. 126-127.

77. G. Hadley, *Linear Programming*, (Reading, Mass.: Addison-Wesley, 1962), pp. 168-170.

78. Barrodale and Young, op. cit., pp. 295-296.

79. R.J. Clasen, *SIMPLE - Linear Programming Simplex Subroutine*, Share Program Library No. SDA 3384, Program RS-LSUB, (Santa Monica: Rand Corp., 1965).

80. R.J. Clasen, *Linear Programming as a Simplex Subroutine*, Rand Report P-3267, (Santa Monica: Rand Corp., 1965).

81. Hammer and Rudeanu, "Pseudo-Boolean Programming," op. cit., pp. 236-241.

82. Hammer and Rudeanu, *Pseudo-Boolean Methods for Bivalent Programming*, op. cit., pp. 23-36.

83. Hammer and Rudeanu, *Boolean Methods in Operations Research*, op. cit., pp. 54-65.

84. Knuth, op. cit., p. 309.

85. Hammer and Rudeanu, *Boolean Methods in Operations Research*, op. cit., pp. 57-59.

86. Ibid., pp. 56-57.

87. Ibid., p. 58.

88. Hammer and Rudeanu, *Pseudo-Boolean Methods for Bivalent Programming*, op. cit., p. 27.

89. Hammer and Rudeanu, "Pseudo-Boolean Programming," op. cit., p. 238.

90. Knuth, op. cit., p. 316.

91. Ibid., pp. 316-317.

92. Ibid.

93. Ibid., pp. 317-318.

94. Lancaster, op. cit.

95. King, op. cit.

96. Salton, op. cit., pp. 283-293.

97. Meetham, op. cit.

98. A. Kent, ed., Electronic Handling of Information: Testing and Evaluation "Relevance Predictability II, by D. Shirey and M. Keefurst," (Washington: Thompson Book Co., 1967), ch. 14.

99. R.H. Shumway, "Contingency Tables in Information Retrieval: An Information Theoretic Analysis," Evaluation of Document Retrieval Systems: Literature Perspective, Measurement, Technical Papers; Part III, Technical Papers. Prepared by Westat Research Inc. for Office of Science Information Service, National Science Foundation Report No. PB. 182-710, (Springfield, Va.: Clearinghouse for Federal Scientific and Technical Information, 1969).

100. Kullback, op. cit.

101. C.R. Hicks, Fundamental Concepts in the Design of Experiments, (New York: Holt, Rinehart and Winston, 1964), pp. 179-185.

102. Ibid., pp. 153-156.

103. W.G. Cochran and G.M. Cox, Experimental Designs, (New York: John Wiley, 1957), pp. 50-53.

104. B.J. Winer, Statistical Principles in Experimental Design, (New York: McGraw-Hill, 1962), pp. 524-529.

105. Cochran and Cox, loc. cit.

106. L.N. Kanal, ed., Pattern Recognition "Property Learning in Pattern Recognition Systems Using Information Content Measurements, by C.W. Swonger," (Washington: Thompson Book Co., 1968), pp. 329-347.

107. Maltz, op. cit.

108. Hammer and Rudeanu, Boolean Methods in Operations Research, op. cit., pp. 65-75.

109. Ibid., pp. 86-90.

110. Barrodale, op. cit.

111. Barrodale and Young, op. cit.

112. Hammer and Rudeanu, <u>Boolean Methods in Operations Research</u>, op. cit., p. 170.

REFERENCES NOT CITED

Anderson, T.W. Introduction to Multivariate Statistical Analysis
New York: John Wiley, 1958, Chapter 6.

Fisher, R.A. "The Use of Multiple Measurements in Taxonomic Problems," Annals of Eugenics, Vol. 7, Part 2 (1936), pp. 179-188.