KNIGHT, James Wayne, 1946-
ON THE SELECTION OF FRACTIONAL FACTORIALS
GIVEN A LIST OF FEASIBLE OBSERVATIONS.
The Ohio State University, Ph.D., 1977
Engineering, industrial

University Microfilms International, Ann Arbor, Michigan 48106

© Copyright by
James Wayne Knight
1977
ON THE SELECTION
OF
FRACTIONAL FACTORIALS
GIVEN A LIST OF FEASIBLE OBSERVATIONS

DISSERTATION

Presented in Partial Fulfillment of the
Requirements for the Degree of Doctor
of Philosophy in the Graduate School of
The Ohio State University

by

James Wayne Knight, B.S., M.Sc.

The Ohio State University
1977

Reading Committee
Professor J. Neuhardt
Professor C. Mount-Campbell
Professor T. Rockwell

Approved by

Adviser
Department of Industrial
and Systems Engineering
I wish to express my grateful thanks to the many individuals who have aided me in this effort.

Foremost I must thank Dr. John B. Neuhardt for his patient guidance throughout my degree program and for his excellent example as a teacher and researcher.

Secondly, I wish to thank Dr. Thomas H. Rockwell for his enthusiastic support of my program, for his ideas, and for the many opportunities he provided me.

The valuable suggestions again of Dr. Rockwell and of Dr. Clark Mount-Campbell, both of whom served on the dissertation reading committee, were most appreciated.

To the people of the Department of Industrial and Systems Engineering and the staff of the Systems Research Group I extend a special thanks.

Ms. Melodee Schmidtlein and Ms. Grace Behn must be acknowledged for their labors in typing an often difficult text.

Lastly, to my parents and my wife's parents I own so much for their unwavering support and assistance. For this I am forever thankful. To Linda, my wife, I thank for the experiences we have shared.
VITA

July 10, 1946 . . . . . . . . . . . . . . . . . . . Born - Columbus, Ohio

1969 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . B.S.E.E., Ohio University, Athens, Ohio

1969-1970 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Research Assistant, Electro-
Science Laboratory, The Ohio State University, Columbus, Ohio

1970-1971 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Research Assistant, Department
of Biophysics and Computer
and Information Science, The
Ohio State University, Columbus, Ohio

1971 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . M.Sc., The Ohio State
University, Columbus, Ohio

1971-1977 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Research Associate, Systems
Research Group, Department of
Industrial and Systems
Engineering, The Ohio State
University, Columbus, Ohio

1977 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Teaching Assistant, Department
of Industrial and Systems
Engineering, The Ohio State
University, Columbus, Ohio

1977 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Assistant Professor of Indus-
trial Engineering, University of Missouri, Columbia, Missouri

PUBLICATIONS

"Guidelines for Field Inventory Procedures (Volumes 1 and 2)," with J. B. Neuhardt, E. A. Whitehurst, and
M. W. Gallogly, Research Foundation Report RF 3535B1, The Ohio State University, Columbus, Ohio 1975.


FIELDS OF STUDY

Major Field: Industrial and Systems Engineering

Studies in Applied Statistics:
Professor John B. Neuhardt

Studies in Human Factors Engineering:
Professor Thomas H. Rockwell
TABLE OF CONTENTS

ACKNOWLEDGMENTS .................................................................................. ii
VITA .......................................................................................................................... iii
LIST OF TABLES ................................................................................................... viii
LIST OF FIGURES ................................................................................................ xi
SYMBOLS ............................................................................................................. xiii

Chapter

I. INTRODUCTION ................................................................................................. 1
   Background ........................................................................................................... 1
   Data Collection Procedures in Controlled Empirical Studies ...................... 1
   Data Collection Procedures With Less Research Control ......................... 3
   Proposed Design Strategy ............................................................................... 6
   Objectives ........................................................................................................... 14
   Scope of Work ................................................................................................... 15

II. LITERATURE REVIEW AND SUPPORTING THEORY .................................. 16
   Introduction ....................................................................................................... 16
   Factorial Structures ......................................................................................... 17
   Order 2^k Modulo 2 Abelian Group Theory .................................................. 21
   Factorial Structure as Abelian Group ............................................................. 23
   Historical Perspective ....................................................................................... 23
   Treatment Group, Effect Group .................................................................... 24
   Construction of Fractions ............................................................................... 26
   Resolution .......................................................................................................... 28
   Upper Bounds on Number of Unique Alias Subgroups ................................ 30
   Systems of Linear Equations ......................................................................... 33
   Relationship Between R, k, p ......................................................................... 35
   Relationships for Wordlengths of the Alias Subgroup .................................. 35
   Inequality Relationship for R, k, p ................................................................. 36
   Maximum Value of k Given R and 2^k-p ......................................................... 37
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary Number Analogy</td>
<td>37</td>
</tr>
<tr>
<td>Propagation of Empty Cells</td>
<td>40</td>
</tr>
<tr>
<td>Development of Supporting Theory and Relationships</td>
<td>41</td>
</tr>
<tr>
<td>Properties of Generator Sets and Alias Subgroups</td>
<td>42</td>
</tr>
<tr>
<td>Feasibility Matrix Cell Dependencies</td>
<td>71</td>
</tr>
<tr>
<td>III. DESCRIPTION OF ALGORITHM</td>
<td>94</td>
</tr>
<tr>
<td>Introduction</td>
<td>94</td>
</tr>
<tr>
<td>Feasibility Matrix</td>
<td>94</td>
</tr>
<tr>
<td>Path of Search Through Feasibility Matrix</td>
<td>96</td>
</tr>
<tr>
<td>Beginning the Search</td>
<td>100</td>
</tr>
<tr>
<td>Determining $k_{min}$</td>
<td>100</td>
</tr>
<tr>
<td>Determining Empty Cell Set and Minimum Column Count</td>
<td>104</td>
</tr>
<tr>
<td>Searching a Cell in the Fractional Factorial Region</td>
<td>105</td>
</tr>
<tr>
<td>Preparation for the Cell Search</td>
<td>105</td>
</tr>
<tr>
<td>Generator Iteration-The Cell Search</td>
<td>108</td>
</tr>
<tr>
<td>Inter-Cell Dependencies</td>
<td>120</td>
</tr>
<tr>
<td>Terminating the Search</td>
<td>124</td>
</tr>
<tr>
<td>IV. PERFORMANCE OF THE ALGORITHM</td>
<td>125</td>
</tr>
<tr>
<td>Efficiency of the Algorithm</td>
<td>125</td>
</tr>
<tr>
<td>Search of Random Data Sets</td>
<td>128</td>
</tr>
<tr>
<td>Highway Intersection Example</td>
<td>131</td>
</tr>
<tr>
<td>V. GENERAL CONSIDERATIONS IN THE USE OF THE ALGORITHM</td>
<td>139</td>
</tr>
<tr>
<td>Solution Approach without Algorithm</td>
<td>139</td>
</tr>
<tr>
<td>Upper Bounds by Cell for Total Enumeration</td>
<td>140</td>
</tr>
<tr>
<td>Strategies to Perform Initial Search</td>
<td>146</td>
</tr>
<tr>
<td>Initial Planning Issues</td>
<td>147</td>
</tr>
<tr>
<td>Partitioning the Data</td>
<td>149</td>
</tr>
<tr>
<td>Partitioning the Extremes</td>
<td>150</td>
</tr>
<tr>
<td>Missing Data</td>
<td>153</td>
</tr>
<tr>
<td>Large Number of Sites per Factor Level Combination</td>
<td>155</td>
</tr>
<tr>
<td>Small Number of Sites</td>
<td>156</td>
</tr>
<tr>
<td>Blocking a Fraction</td>
<td>156</td>
</tr>
<tr>
<td>Chapter</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>VI. SUMMARY AND RECOMMENDATIONS</td>
<td>157</td>
</tr>
<tr>
<td>Summary</td>
<td>157</td>
</tr>
<tr>
<td>Recommendations</td>
<td>159</td>
</tr>
<tr>
<td>Conclusions</td>
<td>160</td>
</tr>
<tr>
<td>APPENDIX</td>
<td></td>
</tr>
<tr>
<td>A Definition of Row Echelon Form</td>
<td>161</td>
</tr>
<tr>
<td>B Enumeration of Cases for Selection of</td>
<td>163</td>
</tr>
<tr>
<td>(p + 1) th Generator</td>
<td></td>
</tr>
<tr>
<td>C Description of Generation of Random</td>
<td>185</td>
</tr>
<tr>
<td>Data Sets</td>
<td></td>
</tr>
<tr>
<td>D Listing of Computer Program</td>
<td>188</td>
</tr>
<tr>
<td>E Tables of Numbers of Unique Subgroups</td>
<td>205</td>
</tr>
<tr>
<td>F Generator Iteration Example</td>
<td>211</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>213</td>
</tr>
</tbody>
</table>
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Run Number, Factors, Design Matrix and Observations for a $2^3$ Factorial</td>
<td>19</td>
</tr>
<tr>
<td>2.2</td>
<td>Run Number, Factors, Design Matrix and Observations for a $2^{3-1}$ Fraction</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>Results of Calculations to Determine Number of Unique Order $2^3$ Subgroups of Order $2^7$ Group Where Each Subgroup Contains All $k$ Factors</td>
<td>31</td>
</tr>
<tr>
<td>2.4</td>
<td>Maximum Values of $k$ Given $R$ and $2^{k-p}$</td>
<td>37</td>
</tr>
<tr>
<td>2.5</td>
<td>Propagation of Empty Cell, $x_{y_1}$</td>
<td>41</td>
</tr>
<tr>
<td>2.6</td>
<td>Minimum Values, $k_3$, for $k$ Given $p$</td>
<td>48</td>
</tr>
<tr>
<td>2.7</td>
<td>Minimum Leading Element Position, $k_3$, for Each Possible Subset Size of REF Generators</td>
<td>51</td>
</tr>
<tr>
<td>2.8</td>
<td>Maximum Row Echelon Form Generator Values, $x'_{gmax_1}$</td>
<td>56</td>
</tr>
<tr>
<td>2.9</td>
<td>Generator Terminating Values, $x_{gmax_1}$</td>
<td>69</td>
</tr>
<tr>
<td>2.10</td>
<td>$H(h)$ and $w(h)$ for All Values of $h$</td>
<td>71</td>
</tr>
<tr>
<td>2.11</td>
<td>Remaining Wordlength Combinations for Distinct $G'$ and a Satisfactory $x_{g_3}$ for $(k,p) = (7,2)$</td>
<td>83</td>
</tr>
<tr>
<td>2.12</td>
<td>For $(k,p) = (7,2)$ All Distinct $G'$ Which Satisfy $| x_{a_i} | \neq 5$ or $6$ and a Satisfactory $x_{g_3}$ and $x_{g_4}$</td>
<td>85</td>
</tr>
<tr>
<td>2.13</td>
<td>For $(k,p) = (7,3)$ All Distinct $G'$ Which Satisfy $| x_{a_i} | \neq 5$ or $6$ and a Satisfactory $x_{g_4}$</td>
<td>87</td>
</tr>
<tr>
<td>2.14</td>
<td>Remaining Wordlength Combinations for Distinct $G'$ and a Satisfactory $x_{g_3}$ for $(k,p) = (8,2)$</td>
<td>88</td>
</tr>
</tbody>
</table>
Table

<p>| 2.15 | All Distinct Generator Sets for ((k,p) = (6,2)) Such That (|x_g| &lt; 5, |x_g| &lt; 5,) (|x_g + x_g| &lt; 6) Plus a Satisfactory (x_g) | Page 90 |
| 3.1 | Resulting Information After First Three Sorts | 102 |
| 3.2 | Inter-cell Dependencies | 121 |
| 4.1 | Search Times and Terminating Cell Results, Grouped by Data Sets, for Random Uniformly Generated Data Sets of Varying Size | 129 |
| 4.2 | Ranking and Coding of Intersection Attributes | 123 |
| 4.3 | Listing of 8 Factor Level Combinations, (x_i), Which Comprise a 2(^6)-3 Fraction and the Number of Times Each (x_i) Occurs in the Intersection Data Base (190 Observations) | 135 |
| 4.4 | Listing of 8 Factor Level Combinations, (x_i), Which Comprise a 2(^7)-4 Fraction with the Number of Times Each (x_i) Occurs in the Intersection Data Base (431 Observations) | 137 |
| 4.5 | Listing of 16 Factor Level Combinations, (x_i), Which Comprise a 2(^6)-2 Fraction with the Number of Times Each (x_i) Occurs in the Intersection Data Base (431 Observations) | 138 |
| 5.1 | Maximum Number of Factors Likely to Be Feasible Given (N) | 144 |
| B.1 | Enumeration of Remaining Distinct Generator Combinations for the (8,3) Cell with Selection of a Suitable Fourth Generator, (x_{g4}) | 165 |
| B.2 | Enumeration of Remaining Distinct Generator Combinations for the (9,2) Cell with Selection of a Suitable Third Generator, (x_{g3}) | 168 |
| B.3 | Enumeration of Remaining Distinct Generator Combinations for the (9,3) Cell with Selection of a Suitable Fourth Generator, (x_{g4}) | 169 |</p>
<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.4</td>
<td>Enumeration of Remaining Distinct Generator Combinations for the (9,4) Cell with Selection of a Suitable Fifth Generator, $x^g_5$</td>
</tr>
<tr>
<td>D.1</td>
<td>Listing of Job Control Language Statements</td>
</tr>
<tr>
<td>D.2</td>
<td>Listing of Algorithm</td>
</tr>
<tr>
<td>E.1</td>
<td>Number of Distinct Order $2^2$ Subgroups</td>
</tr>
<tr>
<td>E.2</td>
<td>Number of Distinct Order $2^3$ Subgroups</td>
</tr>
<tr>
<td>E.3</td>
<td>Number of Distinct Order $2^4$ Subgroups</td>
</tr>
<tr>
<td>E.4</td>
<td>Number of Distinct Order $2^5$ Subgroups</td>
</tr>
<tr>
<td>E.5</td>
<td>Number of Distinct Order $2^6$ Subgroups</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Relationship Between Two Data Bases and Search Algorithm Which Comprise Design Strategy</td>
<td>7</td>
</tr>
<tr>
<td>1.2</td>
<td>A Geometric Representation of the $2^3$ Factor Level Combination for Highway Intersection Attributes</td>
<td>9</td>
</tr>
<tr>
<td>1.3</td>
<td>Feasibility Matrix—All Combinations of Ranked Factors and Observations Arranged in Matrix Form</td>
<td>13</td>
</tr>
<tr>
<td>2.1</td>
<td>Maximum Generator Wordlengths, $</td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td>Feasibility Matrix with Full Factorial Diagonal, Replicated Full Factorial Region, Fractional Factorial Region, and Non-Constructible Region Designated</td>
<td>95</td>
</tr>
<tr>
<td>3.2</td>
<td>The General Sequence in Which Cells are Considered by the Search Algorithm</td>
<td>98</td>
</tr>
<tr>
<td>3.3</td>
<td>Determining $k_{\text{min}}$</td>
<td>101</td>
</tr>
<tr>
<td>3.4</td>
<td>Known Cell Feasibilities, $A(i,j)=1$ and Infeasibilities, $A(i,j)=0$, when $k_{\text{min}}=7$</td>
<td>103</td>
</tr>
<tr>
<td>3.5</td>
<td>Iteration of $x_{g_1}$ for the $p=1$ Case</td>
<td>109</td>
</tr>
<tr>
<td>3.6</td>
<td>Flow Diagram of Generator Iteration for $p\geq 2$ Case</td>
<td>112</td>
</tr>
<tr>
<td>3.7</td>
<td>Final Series of Checks on $G$ to Determine Feasibility</td>
<td>119</td>
</tr>
<tr>
<td>3.8</td>
<td>Implications of the Column, Diagonal and Row Rules</td>
<td>123</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.1</td>
<td>Feasibility Matrix for Highway Intersection Data Base with 190 Observations</td>
<td>134</td>
</tr>
<tr>
<td>4.2</td>
<td>Feasibility Matrix for Highway Intersection Data Base with 431 Observations</td>
<td>136</td>
</tr>
<tr>
<td>5.1</td>
<td>Upper Bounds for Maximum Possible Number of Unique Alias Subgroups by Cell</td>
<td>141</td>
</tr>
<tr>
<td>5.2</td>
<td>Plot of Times to Complete Search Versus Data Base Size with Terminating Cell Value Indicated</td>
<td>143</td>
</tr>
<tr>
<td>5.3</td>
<td>Contours of Times to Complete Search</td>
<td>145</td>
</tr>
<tr>
<td>5.4</td>
<td>Decision Tree for Initial Search</td>
<td>148</td>
</tr>
<tr>
<td>5.5</td>
<td>90% Confidence Interval Width for the Slope Parameter, b, as a Function of $x_{\text{high}} - x_{\text{low}}$</td>
<td>152</td>
</tr>
<tr>
<td>C.1</td>
<td>Generation of Random, Uniform Historical Data Base</td>
<td>187</td>
</tr>
</tbody>
</table>
SYMBOLS

\( a_{ij} \) = the level of the \( j^{th} \) factor for the \( i^{th} \) k-tuple using 0,1 coding

\( a'_{ij} \) = the level of the \( j^{th} \) factor at the \( i^{th} \) observation using +1, -1 coding

\( A \) = order \( 2^p \) alias subgroup of \( X_k \)

\( A^* \) = \{A\} the set of all alias subgroups for a given \( k \) and \( p \)

\( A_s(e_i) \) = \{e_j | e_j = e_i \cdot e_r \text{ for all } e_r \in A\} \), the effect alias set

\( B \) = the binary equivalent of \( x_{g_p} \| x_{g_{p-1}} \| \ldots \| x_{g_2} \| x_{g_1} \)

\( B_i \) = the binary equivalent of \( x_{g_i} \)

\( \| \) = concatenation

\( e_i \) = member of \( E_k \)

\( E \) = empty set, \( E = \{x_{y_1} | n_{y_1} < n_{\text{min}}\} \)

\( \overline{E} \) = the nonempty set, \( \overline{E} = \{x_{z_1} | n_{z_1} \geq n_{\text{min}}\} \)

\( f \) = a family of fractions, \( \{f_1, f_2, f_3, \ldots, f_{2p}\} \)

\( E_k \) = order \( 2^k \) effects group with members \( e_i \)

\( G \) = a set of generators \( (x_{g_1}, x_{g_2}, \ldots, x_{g_p}) \)

\( \overline{G} \) = a \( p \times k \) matrix, 

\[
\begin{pmatrix}
  x_{g_1} \\
  x_{g_2} \\
  \vdots \\
  x_{g_p}
\end{pmatrix}
\]

where \( x_{g_1} \) is a 1xk row vector
$G'$ = a set of generators in row echelon form,
\[(x'g_1, x'g_2, \ldots, x'g_p)\]

$G'$ = a $p \times k$ matrix representation of $G'$

$H$ = a $p \times (k-p)$ matrix being a right partition of $G'$

$k_{\text{max}}$ = maximum number of factors in historical data base

$k$ = number of ranked factors

$k_{\text{min}}$ = the minimum number of ranked factors for which a full factorial is not feasible

$k_3$ = the minimum number of factors necessary for a resolution III design for a $1/2P$ replicate

$L = k \cdot p$, length of the string, $x_{g_p} \parallel x_{g_{p-1}} \parallel \cdots \parallel x_{g_2} \parallel x_{g_1}$

$M$ = number of unique generator sets for an order $2^k$ modulo 2 Abelian group

$m$ = number of replications of a full factorial structure

$n$ = number of potential experimental units for a given combination of factor levels

$n_{\text{min}}$ = minimum number of potential experimental units required for each combination of factor levels

$N$ = the number of unique order $2^p$ subgroups of an order $2^k$ group

$p$ = the exponent of $\frac{k}{2}$, representing the degree of fractional replication

$p_{\text{min}}$ = minimum possible value of $p$ for a given set of conditions

$q$ = $k - p$

$q_i$ = $i^{th}$ member of historical data base

$Q = \{q_i\}$, the historical data base
$R = \text{resolution, designated with a Roman numeral}$

$R_{\text{max}} = \text{the maximum possible resolution given } k, p$

$x_i = (a_{i1}, \ldots, a_{i2}, a_{i3}), \text{a binary } k\text{-tuple}$

$x_{a_i} = \text{ith member of alias subgroup, } A$

$x_{g_i} = \text{ith member of the generator set } G$

$x_{g'_i} = \text{ith member of the row echelon form generator set, } G'$

$x_{y_i} = \text{ith member of the empty set, } E$

$x_{z_i} = \text{ith member of the nonempty set, } E$

$X_k = \text{full factorial structure in } k \text{ factors}$
CHAPTER I
INTRODUCTION

1.1 BACKGROUND

1.1.1 Data Collection Procedures in Controlled Empirical Studies

In controlled empirical scientific inquiry, the researcher has a question to be investigated. To conduct the study the variables of concern are divided into two classes: 1) those variables that the researcher manipulates or controls, designated as the independent variables (i.e., independent because their values do not depend upon the values of other variables), and 2) those variables which are measured or observed subsequent to the setting of the independent variables, thus designated as dependent variables (i.e., dependent upon the values of the independent variables). In simplest terms conducting the experiment, then, consists of changing the independent variables in some systematic manner and observing the changes, if any, in the dependent variables. Based upon these observations the researcher then arrives at conclusions regarding the existence of cause-effect relationships or patterns of covariation between the independent and dependent variables.
Often to aid the explanation of what has been observed, the researcher either explicitly or implicitly assumes a model which relates the observed values of the dependent variable to the controlled values of the independent variables. Let the general form of this model be represented as

\[ E(y_i) = f(x_{i1}, x_{i2}, \ldots, x_{ik}) \quad i=1, 2, \ldots, n \quad (1.1) \]

where:

- \( y_i \) is the \( i \)th observation of the dependent variable,
- \( E(y_i) \) is the expected value of the dependent variable,
- \((x_{i1}, x_{i2}, \ldots, x_{ik})\) is the combination of values of the independent variables for which the dependent variable is observed, and
- \( f(x_{i1}, x_{i2}, \ldots, x_{ik}) \) is a function, usually assumed to be appropriate, which relates the values of the independent variables to the expected value of the dependent variable.

For discussion purposes equation (1.1) is designated the "general model".

The experimenter's ability to control the values of the independent variables in the controlled empirical study has two important consequences. First, if the set of independent variable values for the \( n \) observations,

\[
\begin{align*}
(x_{11}, x_{12}, \ldots, x_{1k}) & \quad \text{(observation 1)} \\
(x_{21}, x_{22}, \ldots, x_{2k}) & \quad \text{(observation 2)} \\
\cdots & \\
(x_{n1}, x_{n2}, \ldots, x_{nk}) & \quad \text{(observation } n) 
\end{align*}
\]  

(1.2)
are properly selected, then the effect of each independent variable upon the dependent variable can be unambiguously estimated.

The second consequence is that the direction of cause-effect relationships can be established. When the researcher has direct control over the independent variables, he can control the chronological sequence of first setting the independent variable values and then observing the dependent variable value. If the dependent variable changes systematically, then a cause-effect relationship has been identified. The validity of this relationship assumes that the researcher has accounted for all other factors, in addition to the independent variables, which could have affected the dependent measure.

1.1.2 Data Collection Procedures with Less Researcher Control

Because of the advantages of controlled empirical study, it is the preferred method when usable. However, there are many empirical studies in which circumstances dictate that some aspects of the controlled approach must be restricted in order to investigate the question [Kish (1959), Runkel and McGrath (1972)]. The issue being addressed in this work concerns what design strategy can be employed when the researcher does not have direct control over the independent variables. The following are three
examples demonstrating this issue.

In human factors laboratory experiments, characteristics of the subjects are often treated as independent variables. Frequently these characteristics cannot be directly controlled. Examples of commonly considered attributes are: sex; amount of previous experience, learning or skill; age; body measurements; and degree of acclimatization to some environmental condition such as heat stress.

In field experiments and field studies (Runkel and McGrath, 1972) the researcher investigates relationships between attributes of the field setting and the dependent measure. As an example consider the research question: Do longitudinal deceleration patterns of automobiles approaching highway intersections vary systematically from intersection to intersection? To conduct the study, unobtrusive observation of decelerating vehicles on approaches to highway intersections could be the method of investigation. The attributes of each intersection would be the independent variables. These might include: whether the intersection was signalized or unsignalized, the average approach velocity, the number of lanes, the average number of cars using the intersection from 6:30 a.m. to 6:30 p.m., and the total number of accidents during the previous five years. Each intersection can be represented by an ordered k-tuple of independent variable values, \((x_{i1}, x_{i2}, \ldots, x_{ik})\). The dependent measure might be the average deceleration exhibited
by all stopping vehicles in the 250 foot to 150 foot interval prior to the intersection.

Sample surveys are a third example of when the investigator often does not have direct control over the independent variables. The objective of a sample survey is to assess or describe (using a questionnaire, say) the attitudes and preferences of people belonging to specific populations. Based upon results of the sample, inferences are made concerning the population. The attributes of the population play the role of the independent variables. Frequently considered attributes might include: income, political party affiliation, sex, educational background, religion, home ownership or age. The responses to the survey's questions are the dependent measure.

In the preceding three examples the researcher may not have direct control over the independent variables in that the human factors subject, the highway intersection, or the population member must be identified which possesses (or nearly so) the desired attributes. Only by this indirect manner of experimental unit selection can the independent variables be controlled. Control does not exist in the traditional, direct sense.

Reiterating the discussion of the controlled empirical study but now from a negative point of view, there are two major implications when the researcher must select "units" possessing attributes which are considered as independent
variables. First, if no attempt is made to obtain sets of independent variable values, \((x_{i1}, x_{i2}, \ldots, x_{ik})\), following some type of experimental design, the information content of the data would very likely be unsatisfactory. That is the data would not allow the effects of the independent variables to be unambiguously estimated. Secondly, when more than two or three factors are being indirectly controlled by means of experimental unit selection, it can become extremely difficult to locate these units possessing the necessary combinations of attributes. However, studies which use unit selection control, often need to consider as many unit attribute factors as possible because it is not known (a priori) which two or three may be the most important.

Therefore a method is needed which would allow studies, that cannot directly control the independent variables, to consider more factors than is often the case without incurring excessive costs to obtain the necessary experimental units. This dissertation develops a computer-aided design methodology which addresses the above problem as discussed in the following section.

1.2 PROPOSED DESIGN STRATEGY

The two requirements for selection of experimental units, when the units' attributes are considered as independent variables and are not directly controlled, are:

1. to simultaneously consider the maximum possible number of attributes of the experimental units, and
2. to select experimental units such that the resulting data of $y_i$ and $(x_{i1}, x_{i2}, \ldots, x_{ik})$ values allow the effects of attributes to be unambiguously estimated.

The proposed design strategy, to be described next, satisfies the above two requirements.

Figure 1.1 displays the three components of the design strategy. For input, the design strategy requires the availability of a historical data base which contains the attribute values of potential experimental units.

![Diagram](image)

Figure 1.1--Relationship Between Two Data Bases and Search Algorithm Which Comprise Design Strategy
Such data bases are very prevalent in many industries, throughout government, and among social institutions. Therefore the availability of this data is often not a problem.

The historical data base is obtained by transforming each k-tuple, \((x_{i1}, x_{i2}, \ldots, x_{ik})\), of experimental unit attribute values into a binary k-tuple,\
\[(a_{i1}, a_{i2}, \ldots, a_{ik})\]  
(1.3)
where \(a_{ij}\) equals 0 or 1, representing the \(j^{th}\) attribute level for the \(i^{th}\) unit. For quantitative attributes the 0, 1 coding represents low and high values. For qualitative attributes the coding could represent, for example, "not present, present", or "yes, no" responses. It is assumed that the two-level coding is adequate (e.g., screening experiments or exploratory studies [John (1971), Mendenhall (1968)]).

With each attribute coded at two levels, then for \(k\) attributes, there are \(2^k\) possible k-tuple combinations. Figure 1.2 illustrates, for three attributes, the eight possible combinations of attributes of intersections for the highway example.

The factorial data base of Figure 1.1 contains, for each possible k-tuple combination, the number of units in the historical data base with that combination. Therefore the factorial data base has \(2^k\) paired entries with each consisting of a binary k-tuple and its corresponding frequency
Figure 1.2—A Geometric Representation of the $2^3$ Factor Level Combination for Highway Intersection Attributes
of occurrence.

The set of $2^k$ possible factor or attribute level combinations for $k$ factors is designated a full factorial. If $2^k$ experimental units which comprise a full factorial structure are selected for study, then the effect of each attribute and every possible interaction effect of two or more attributes is unambiguously estimable. Therefore a full factorial selection pattern satisfies the estimability requirement for selection of experimental units.

If the total number of attributes, $k$, is large, then the total number of experimental units required by the full factorial structure becomes very large (e.g., if $k = 6$, then $2^k = 64$; if $k = 8$, then $2^k = 256$). Therefore as $k$ increases, an alternate structure which requires fewer experimental units is needed for selecting units. An orthogonal fractional factorial structure satisfies this need. Fractional factorials (Box and Hunter, 1961) consist of a fraction (power of two) of the full factorial (i.e., $1/2$, $1/4$, $1/8$, ..., $1/2^p$) but still allow all main effects due to attributes to be estimated (and also some interaction effects depending upon the particular fractional factorial structure). Therefore if experimental units are selected according to either a full or fractional factorial structure, the estimability condition is satisfied.

To determine the maximum number of attributes or factors that can be simultaneously considered, a computerized
search algorithm has been developed. The algorithm systematically searches the factorial data base to ascertain the maximum number of factors that can be considered using a given set of experimental units which collectively satisfy a full or fractional factorial structure.

The search algorithm requires the researcher to rank the factors by their (a priori) judged importance or probable strength of relation with the dependent measure. Then the algorithm begins the search task by first determining the largest full factorial structure for which all of the necessary factor level combinations have matching experimental units in the data base. A structure is denoted as feasible if all matching units exist.

A graphical means of representing the results of the search is the feasibility matrix shown in Figure 1.3. The feasibility matrix consists of all possible combinations of ranked factors \( k \) and numbers of observations \( 2^q \). Then a given cell, \( (k,2^q) \), of the matrix is feasible if there exists at least one corresponding structure for that cell which is feasible.

As shown in Figure 1.3, the ordinate of the feasibility matrix consists of the number of ranked factors being considered which can vary from one to as many as ten. The abscissa is the number of observations, expressed as a power of two, varying from 2 to \( 2^{10} \).
The cells of the feasibility matrix can be classified into four types. The first type are those cells which make up the full factorial diagonal. For these cells the number of observations equals the size of the full factorial structure, that is

\[ 2^q = 2^k \]

The full factorial diagonal separates the replicated full factorial region from the fractional factorial region. In the former region a cell represents a replication of the full factorial, say \( r \) times, such that

\[ 2^q = r \cdot 2^k \]

In the fractional factorial region, \( 1/2^p \) replicates of full factorial structures are represented and

\[ 2^q = 2^{k-p} \]

Lastly, the nonconstructible region of Figure 1.3 consists of those combinations of \( k \) and \( 2^q \) for which orthogonal fractional factorials do not exist (Box and Hunter, 1961).

Therefore with respect to the feasibility matrix, determining the largest feasible full factorial corresponds to the search algorithm finding the highest feasible cell on the full factorial diagonal. Once obtained, then the algorithm begins to search for feasible fractional factorials (i.e., mapping the fractional factorial of Figure 1.3). Because for \( k \) greater than six the number of potential fractional factorials is very large (e.g., for \( k = 7 \) an upper bound is 4811 potential structures), searching for feasible
Figure 1.3—Feasibility Matrix—All Combinations of Ranked Factors and Observations Arranged in Matrix Form
fractional factorials is the difficult aspect of the search task.

Therefore, in summary, the proposed design strategy requires the availability of an historical data base containing 0, 1 coded attribute values of potential experimental units. The factorial data base consists of the number of units in the historical data base corresponding to each of the $2^k$ factorial attribute combinations. Using full factorial and fractional factorial structures, a computerized search algorithm searches the factorial data base determining the largest number of factors that can be simultaneously considered while satisfying the estimability requirement.

1.3 OBJECTIVES

This dissertation has three major objectives. The first objective is:

1. To develop a computer aided design strategy that selects experimental or sampling units based upon their attribute values, which are coded at two levels, such that a larger number of attributes can be simultaneously yet unambiguously considered in a study than would be possible by conventional strategies.

Usually there are many experimental or sampling unit attributes of interest to a researcher. The first objective concerns the problem that the researcher is very often limited to considering only a few such attributes because there is no practical, economical method for selecting units which would allow a larger number of attributes to be considered.
The second objective follows from the first and is:

2. To develop a computerized algorithm which exhaustively searches for the first feasible solution of the maximum number of attributes that can be considered given a ranking of factors and a coded historical data base.

Lastly, the third objective is a refinement of the second and is:

3. To develop a search algorithm that is efficient in that cases are eliminated from the search based upon partial search results.

1.4 SCOPE OF WORK

Four major limitations are associated with the proposed design strategy and search algorithm. They are:

1. The maximum number of factors that will be considered is limited to ten (although the basic logic is not restricted to such a number).

2. Each factor must be partitioned into two levels or classifications.

3. Only orthogonal fractional factorials for which, minimally, main effects can be estimated independent of other main effects, are considered as potential feasible solutions.

4. When a feasible fractional factorial is determined for a given number of factors, $k$, and a specified fraction, $p$, further searching ceases for that combination of $k$ and $p$. 
2.1 INTRODUCTION

The key to the design strategy is the search algorithm and how well it performs. Performance is judged by the size of the problem that can be solved, where "size" is represented by the number of ranked factors, k, to be considered. The largest value of k to be successfully considered by the algorithm depends upon the specific characteristics of the historical data base under study and upon the efficiency of the algorithm.

The principle task of the algorithm is to exhaustively search each cell in the fractional factorial region of the feasibility matrix. However, what constitutes an exhaustive search is not clear. Secondly because each fractional factorial can, in general, be specified with multiplicity, there is substantial opportunity for redundancy to exist in the search. Hence a measure of efficiency of the search is the minimization of redundancy while maintaining exhaustiveness. In addition the algorithm is more efficient if it can implicitly eliminate cases without explicit examination.

The remainder of this chapter reviews the literature and presents the notation. It also develops the theory on
which the algorithm is based.

2.2 FACTORIAL STRUCTURES

As a point of departure, a simple full factorial and a fractional factorial (hereafter referred to as a fraction) are first discussed. The presentation is patterned after that of Box and Hunter (1961).

Consider three factors 1, 2, 3 each coded at two levels 0, 1. If the factors are quantitative, the coding represents low and high values. If some factors are qualitative, then the coding could represent, for example, "not present, present", or "yes, no" responses.

Observation of a dependent measure, y, for all possible factor level combinations comprises a $2^3$ factorial. The linear model usually assumed for the expected value of the $i^{th}$ observation is:

$$E(y_i) = b_0 + b_1 a'_i + b_2 a'_i + b_3 a'_i + b_{12} a'_i + b_{13} a'_i + b_{23} a'_i + b_{123} a'_i$$

where:

- $y_i$ is the $i^{th}$ observed value of the dependent measure,
- $b_0$ is the average response,
- $b_j$ is the effect of the $j^{th}$ factor,
- $b_{j...k}$ is the interaction effect of factors $j...k$,
- $a'_ij$ is the level of the $j^{th}$ factor at the $i^{th}$ observation, and
\( a'_{i,j...k} \) is the level of the \( j...k^{th} \) interaction at the \( i^{th} \) observation.

Table 2.1 contains the factor levels and design matrix for the \( 2^3 \). The design matrix is (orthogonally) coded in +1 and -1 with the transformation from 0,1 coding being
\[
a'_{ij} = 2a_{ij} - 1. \tag{2.2}
\]
The levels of the interaction terms in the design matrix are obtained as the products of the levels of the corresponding factors. For example, the level of the 123 interaction term for the first observation is
\[
a'_{1,123} = (a'_{1,1}) (a'_{1,2}) (a'_{1,3}) = (-1) (-1) (-1) = -1.
\]
The estimate of the \( j...k \) effect can be obtained by using the calculation
\[
\text{j...k effect estimate} = 1/8 \sum_{i=1}^{8} a'_{i,j...k} y_i \tag{2.3}
\]
For a one-half orthogonal replicate of three factors in four observations, a \( 2^3-1 \) fraction, the experimenter must decide which four of the eight possible factor-level combinations are to be used. Actually the decision is simpler than this since only the effects to be estimated must be specified. Using the rationale that the higher the order of interaction the less likely the effect will be important, the experimenter often chooses the 123 interaction effect as the least "important". Then four factor-level combinations are selected, all of which have either the 123 effect at a
### TABLE 2.1
Run Number, Factors, Design Matrix and Observations for a $2^3$ Factorial

<table>
<thead>
<tr>
<th>Run Number</th>
<th>Factors 1 2 3</th>
<th>Design Matrix</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 0 0</td>
<td>1 -1 -1 -1 1 1 1 1 -1</td>
<td>$y_1$</td>
</tr>
<tr>
<td>2</td>
<td>0 0 1</td>
<td>1 -1 -1 1 1 -1 -1 1 1</td>
<td>$y_2$</td>
</tr>
<tr>
<td>3</td>
<td>0 1 0</td>
<td>1 -1 1 -1 -1 1 -1 1 1</td>
<td>$y_3$</td>
</tr>
<tr>
<td>4</td>
<td>0 1 1</td>
<td>1 -1 1 1 -1 -1 1 -1 1</td>
<td>$y_4$</td>
</tr>
<tr>
<td>5</td>
<td>1 0 0</td>
<td>1 1 -1 -1 -1 1 1 1 1</td>
<td>$y_5$</td>
</tr>
<tr>
<td>6</td>
<td>1 0 1</td>
<td>1 1 -1 1 -1 1 -1 -1 1</td>
<td>$y_6$</td>
</tr>
<tr>
<td>7</td>
<td>1 1 0</td>
<td>1 1 1 -1 1 -1 1 -1 1</td>
<td>$y_7$</td>
</tr>
<tr>
<td>8</td>
<td>1 1 1</td>
<td>1 1 1 1 1 1 1 1 1 1</td>
<td>$y_8$</td>
</tr>
</tbody>
</table>

### TABLE 2.2
Run Number, Factors, Design Matrix and Observations for a $2^3-1$ Fraction

<table>
<thead>
<tr>
<th>Run Number</th>
<th>Factors 1 2 3</th>
<th>Design Matrix</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0 0 1</td>
<td>1 -1 -1 1 1 -1 -1 1 1</td>
<td>$y_2$</td>
</tr>
<tr>
<td>3</td>
<td>0 1 0</td>
<td>1 -1 1 -1 -1 1 1 -1 1</td>
<td>$y_3$</td>
</tr>
<tr>
<td>5</td>
<td>1 0 0</td>
<td>1 1 -1 -1 -1 1 -1 1 1</td>
<td>$y_5$</td>
</tr>
<tr>
<td>8</td>
<td>1 1 1</td>
<td>1 1 1 1 1 1 1 1 1 1</td>
<td>$y_8$</td>
</tr>
</tbody>
</table>
high level (+1) or at a low level (-1).

Table 2.2 contains the four observations having 123 at a high level. Therefore, the column in the design matrix under 123 consists entirely of 1's such that no estimate of a 123 effect is possible.

There remain six effects plus an average response to be estimated from the Table 2.2 data, but only four observations are available. Examining the columns of the design matrix of Table 2.2, columns 1,23 and 2,13 and 3,12 are identical. Therefore an estimate of effect 1 is equal to the estimate of 23, the estimate of 2 is equal to that of 13, etc. (i.e., these effects are confounded). The effect pairs (1,23), (2,13) and (3,12) are said to be aliases of each other. Using equation (2.3) but for only four observations, the four estimates are:

\[
\text{average} = \frac{1}{4} (y_2 + y_3 + y_5 + y_8) \\
1 + 23 = \frac{1}{4} (-y_2 - y_3 + y_5 + y_8) \\
2 + 13 = \frac{1}{4} (-y_2 + y_3 - y_5 + y_8) \\
3 + 12 = \frac{1}{4} (y_2 - y_3 - y_5 + y_8)
\]

In using the estimates the experimenter must decide if the two-way interaction confounded with the main effect is negligible with respect to the main effect. If it is not negligible, then the estimates must be treated as the sum of the two effects. Lastly, it is important to note in Table 2.2 that, for each pair of factors, a $2^2$ factorial exists.
2.3 ORDER $2^k$ MODULO 2 ABELIAN GROUP THEORY

In the development of the search algorithm some elementary concepts from finite Abelian group theory are employed. These concepts are presented here in form and notation consistent with the rest of this work. A general discussion of the theory can be found in Carmichael (1937) and in Ledermann (1961).

Two types of Abelian groups are considered where "type" is determined by the rule of composition for elements of the group. An order $2^k$ modulo 2 Abelian group of the additive type, $X_k$, is a set of $2^k$ binary $k$-tuples,

$$x_i = (a_{ik}, a_{i(k-1)}, \ldots, a_{i2}, a_{i1}),$$

(2.4)

(where $a_{ij} = 0, 1$) with the properties for any elements $x_i, x_j, x_q \in X_k$:

1. The rule of composition, denoted by "+$", is

$$(x_i + x_j) = (a_{ik} + a_{jk} \pmod{2}), \ldots,$$

$$a_{i2} + a_{j2} \pmod{2}, a_{i1} + a_{j1} \pmod{2}),$$

(2.5)

2. $x_i + x_j \in X_k$,

(2.6)

3. $x_i + (x_j + x_q) = (x_i + x_j) + x_q$,

4. $X_k$ contains a single identity element,

$$x_0 = (0, \ldots, 0)$$

such that $x_i + x_0 = x_i$,

5. $x_i + x_i = x_0$, and
6. \( x_i + x_j = x_j + x_i \).

The second type is an order \(2^k\) modulo 2 Abelian group of multiplicative type, \(E_k\), consisting of a set of \(2^k\) elements \(e_i\), where each \(e_i\) is any combination of none, to \(k\) numerals, 1, 2, ..., \(k\) with the properties:

1. The rule of combination, denoted by ".", is numeral-by-numeral multiplication with all exponents modulo 2, e.g.,

\[
1234 \cdot 347 = 123^2 \pmod{2} 4^2 \pmod{2} 7 = 127
\]

If \(a, b, c \in E_k\), then:

2. \(a \cdot b \in E_k\),

3. \(a \cdot (b \cdot c) = (a \cdot b) \cdot c\)

4. \(E_k\) contains a single identity element, \(I\), such that \(a \cdot I = a\),

5. \(a \cdot a = I\), and

6. \(a \cdot b = b \cdot a\).

A subgroup, \(A\), of \(X_k\), is an order \(2^p\) group, \(p < k\), such that if \(x_i \in A\) then \(x_i \in X_k\).

A generator set*, \(G = (x_{g_1}, x_{g_2}, ..., x_{g_k})\), for \(X_k\) is a set of \(k\) elements \(x_{g_i} \in X_k\) with the property that

* Elements of \(G\) are always written with subscripted "g" subscripts.
\[ X_k = \{ x_j \mid x_j = \sum_{i=1}^{k} A_i x_{g_i}, A_i = 0 \text{ or } 1 \} \]  \hspace{1cm} (2.8)

A generator set, \( G = (e_{g_1}, e_{g_2}, \ldots, e_{g_k}) \), for the multiplicative group, \( E_k \), has the property

\[ E_k = \{ e_j \mid e_j = \prod_{i=1}^{k} A_i \} \]  \hspace{1cm} (2.9)

The number of unique generator sets for \( X_k \) is denoted by \( M \) and equals (Carmichael, 1937)

\[ M = \frac{(2^k-1) (2^{k-2}) \ldots (2^{k-2^{k-1}})}{k!} \]  \hspace{1cm} (2.10)

Lastly, the number of unique order \( 2^P \) subgroups of an order \( 2^k \) group is denoted by \( N \) and equals (Carmichael, 1937)

\[ N = \frac{(2^k-1) (2^{k-2}) (2^{k-2^2}) \ldots (2^{k-2^P-1})}{(2^P-1) (2^{P-2}) (2^{P-2^2}) \ldots (2^{P-2^P-1})} \]  \hspace{1cm} (2.11)

### 2.4 FACTORIAL STRUCTURE AS ABELIAN GROUP

#### 2.4.1 Historical Perspective

For a full factorial structure of order \( 2^k \) with each factor at two levels, the set of treatment combinations satisfies the properties of an order \( 2^k \) modulo 2 Abelian group. Fisher (1942) was the first to use the Abelian group property to describe the confounding of effects with blocks when the full factorial is run as a series of blocks. Finney (1945) extended Fisher's work to develop the concept of a fractional replicate and its alias structure. Since these two pioneering works, many others have extended the area of fractional replicates. Kempthorne (1952) covers
several topics in the area, including a development of the general case. Box and Hunter (1961) provide a very complete coverage of fractional replicates for the two level case, oriented toward applications of the ideas. Much theoretical work has been done on many different variations of factorial structures. Addelman (1972, 1963) reviews some of these developments.

2.4.2 Treatment Group, Effect Group

As stated above, the treatment combinations of a full factorial satisfy the properties of an Abelian group. Letting the treatment combinations be represented by binary k-tuples, \( x_i \) (see equation 2.4), they satisfy the properties of an order \( 2^k \) modulo 2 Abelian group of the additive type. Denote this group by \( X_k \).

Similarly the set of effects, represented by numeral elements, \( e_i \), together with the average response, represented by \( I \), has the properties of an order \( 2^k \) modulo 2 Abelian group of multiplicative type. Designate this group by \( E_k \).

The definition of inner product for \( X_k \) is now needed to describe a useful relationship that exists between \( X_k \) and \( E_k \). Given \( x_i, x_j \in X_k \), then the inner product, \( x_i \cdot x_j \), of the two binary k-tuples is

\[
 x_i \cdot x_j = \sum_{q=1}^{k} a_{iq} a_{jq} \text{ (modulo 2)} . \tag{2.12}
\]
(NOTE: No confusion should arise between the inner product specification and the multiplicative rule of combination for \( E_k \), because the former is only applied to \( X_k \) and not \( E_k \).)

For any \( x_i \in X_k \), other than the identity \( x_0 \), \( x_i \cdot x_j \) (over all \( x_j \), \( x_j \in X_k \)) divides \( X_k \) into two sets of order \( 2^{k-1} \) each. Thus to each \( x_i \) there corresponds a particular contrast which estimates an effect by contrasting the one order \( 2^{k-1} \) set with the other. Also the halvings corresponding to any two \( x_i \) are orthogonal (Fisher, 1942). Therefore, using the inner product, a one-to-one correspondence can be set up between \( X_k \) and \( E_k \). It turns out that \( x_i \) with \( a_{ia} = a_{ib} = \ldots = a_{ic} = 1 \) corresponds to effect \( e_i = ab \ldots c \). For example with \( k = 3 \) the corresponding \((x_i, e_i)\) pairs are:

\[
\begin{array}{ll}
(001,1) \\
(010,2) \\
(100,3) \\
(011,12) \\
(101,13) \\
(110,23) \\
(111,123)
\end{array}
\]

Therefore, because of the one-to-one correspondence and because both \( X_k \) and \( E_k \) are order \( 2^k \) modulo 2 Abelian groups, \( X_k \) and \( E_k \) are simply isomorphic (Finney, 1945).

Often it is convenient to work with \( X_k \) if the discussion concerns construction of fractions and \( E_k \) if the issues concern effects or aliases of effects. The simple isomorphism of \( X_k \) and \( E_k \) allows very straightforward transition
from the notation of one group to the other whenever the subject matter warrants such a change.

2.4.3 Construction of Fractions

To construct a $2^{k-p}$ fraction the researcher must first select the $p$ ($p<k$) effects, $e_i \in E_k$, which are of least interest to him. As suggested in the example of Section 2.2, these might consist of the higher order interaction effects. If the set of effects $(e_1, e_2, \ldots, e_p)$ are selected, then the corresponding set of treatment combinations, denoted $G$, where

$$G = (x_{g_1}, x_{g_2}, \ldots, x_{g_p})$$ (2.13)

is designated as the generator set*. A requirement of $G$ is that the $x_{g_i}$ must be independent (i.e., there can be no values of $A_j = 0, 1$ for which

$$x_{g_i} = \sum_{j=1}^{p} A_j x_{g_j}$$

for any $i$).

To construct the $2^{k-p}$ fraction, begin by using $x_{g_1}$ to divide $X_k$ into two sets of $2^{k-1}$ elements each. The set to which an $x_i$ belongs is determined by $x_i \cdot x_{g_1}$ (i.e., if $x_i \cdot x_{g_1} = 0$, $x_i$ goes in one set; and if $x_i \cdot x_{g_1} = 1$,

* Elements of $G$ are always written with subscripted "g" subscripts.
$x_i$ goes in the other set). Next, with respect to one of these sets, divide it in half based upon the values of $x_i \cdot x_{g_2}$. Continue in this manner for all of the remaining generators $x_{g_3}, x_{g_4}, \ldots, x_{g_p}$. The result is a $1/2^p$ replicate of $X_k^*$, denoted $f_i$, consisting of $2^{k-p}$ elements $x_i$ which have the property that either $x_i \cdot x_{g_j} = 0$ or $x_i \cdot x_{g_j} = 1$ for all $i = 1, \ldots, 2^{k-p}$ and $j = 1, \ldots, p$. (Note: Other equivalent methods exist for constructing fractions, see for example Kempthorne (1952) or Box and Hunter (1961).)

It can be shown that for any member of the set $A$, where

$$A = \{x_{a_j} \mid x_{a_j} = \sum_{i=1}^{p} A_i x_{g_i}, \text{ for all } A_i = 0,1, x_{g_i} \in G\} \quad (2.14)$$

and for all $x_i \in f_i$, that $x_i \cdot x_{a_j} = 0$ or 1. This means that $e_{a_j}$ cannot be estimated. The set $A$ is designated the alias subgroup, an order $2^p$ subgroup of $X_k^*$. The corresponding subgroup of $E_k$ is also denoted by $A$. As an example of effects that cannot be estimated, suppose $a, b, c \in E_k$ correspond to $x_{g_1}, x_{g_2}, x_{g_3}$. Then not only would estimates not be available for effects $a, b, c$ but also for effects $a \cdot b, a \cdot c, b \cdot c, a \cdot b \cdot c$.

Actually the selection of a generator set, $G$, specifies not one but a set of $2^p$ fractions of $2^{k-p}$ observations each. This set denoted by $f$, is called the family of fractions for

* Elements of $A$ are always written with subscripted "a" subscripts.
G. Letting \( f_i \) represent a member of \( f \), then each \( f_i \) can be specified as

\[
f_i = \{ x_i \mid x_i \cdot x_{g_1} = c_1, x_i \cdot x_{g_2} = c_2, \ldots, x_i \cdot x_{g_p} = c_p \} \tag{2.15}
\]

where \( c_1, c_2, \ldots, c_p \) assume one of \( 2^p \) possible combinations of 0,1. Hence \( f \) represents a mutually exclusive and collectively exhaustive partitioning of \( X_k \) into \( 2^p \) subsets, \( f_i \), each of order \( 2^{k-p} \).

In general the set \( G \) is not the only set of generators for \( A \). From equation 2.10, \( A \) has \( M \) sets of generators, \{\( G \)\}. Therefore for every alias subgroup, \( A \), there corresponds a unique family of fractions, \( f \), but for every \( f \) there corresponds more than one \( G \).

Lastly, for each effect, \( e_i \), which is estimable, there is a set of \( 2^p \) effects with which \( e_i \) is confounded. This set is designated the effect alias set for \( e_i \), denoted \( A_s(e_i) \), and equals

\[
A_s(e_i) = \{ e_j \mid e_j = e_i \cdot e_q \text{ for all } e_q \in A \} . \tag{2.16}
\]

Therefore for a \( 2^{k-p} \) fraction, each of the \((2^{k-p}-1)\) effect estimates represents a linear combination of \( 2^p \) effects.

### 2.4.4 Resolution

Resolution is a measure of the extent to which effects are confounded in fractional factorials. Before defining resolution, the definition of wordlength is needed. For \( x_i = (a_i k, \ldots, a_i l) \in X_k \), wordlength, denoted \( \| x_i \| \), is defined as
Therefore \( \| x_i \| = \sum_{j=1}^{k} a_{ij} \) \hspace{1cm} (2.17)

Therefore \( \| x_i \| \) is simply the number of 1's in the binary \( k \)-tuple. For \( e_i \in E_k \), \( \| e_i \| \) is the number of numerals appearing in the effect.

Box and Hunter (1961) originated the term resolution, defining it as follows: "A design of resolution \( R \) is one in which no \( p \) factor effect is confounded with any other effect containing less than \( R-p \) factors" (p.319). Therefore resolution, denoted by \( R \), equals the minimum wordlength of the alias subgroup, \( A \). For example consider the \( 2^{6-3} \) fraction with \( e_{g_1} = 124 \), \( e_{g_2} = 135 \), and \( e_{g_3} = 236 \). Then \( A \) equals:

1
124
135
236
2345
1346
1256
456

Therefore \( R = III \) (in the literature resolution is often recorded as a Roman numeral, that convention is followed here). Using equation (2.16), then for the above example the effect alias set for main effect 1 is

\( A_S(1) = (1, 24, 35, 1236, 12345, 346, 256, 1456). \)

Hence the main effect estimate for factor 1 is confounded with two two-way and other higher order interactions effects.
As can be reasoned from the above example, $R=II$ fractions are probably not of interest because main effects are confounded with main effects. In the literature $R=III, IV,$ and $V$ fractions have received the most attention (Addelman, 1972). For $R=IV$, main effects are confounded with three-way and higher interactions. For $R=V$, main effects are confounded with four-way and higher interactions, while two-way interactions are confounded with three-way and higher interactions. For $R>VI$ the number of observations required is often prohibitively large and wasteful in the sense that only a small percentage of the degrees of freedom are used to estimate main effects and two-way interaction (Daniel, 1962).

2.4.5 Upper Bounds on Number of Unique Alias Subgroups

Brownlee, Kelly, and Loraine (1948), using equation (2.11), demonstrate a technique to calculate the number of unique subgroups of order $2^p$ of a group of order $2^k$ where each subgroup contains all $k$ factors (i.e., each of the $k$ factors appears at least once in the subgroup). Their example of $k=7$ and $p=3$ is repeated here. The problem is to calculate the number of unique subgroups of order $2^3=8$ of an order $2^7$ group where each subgroup contains all seven factors.

Table 2.3 is used to summarize the calculations. Beginning in the left most column of Table 2.3 and using equation (2.11) the number of subgroups of order $2^3$ of an
Table 2.3

Results of Calculations to Determine Number of Unique Order $2^3$ Subgroups of Order $2^7$ Group Where Each Subgroup Contains All $k$ Factors

<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Total Number of Subgroups of Order 8</th>
<th>Number of Subgroups Having at Least 4 Factors</th>
<th>Number of Subgroups Having at Least 5 Factors</th>
<th>At Least 6 Factors</th>
<th>At Least 7 Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>155</td>
<td>145</td>
<td>90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1395</td>
<td>1375</td>
<td>1210</td>
<td>670</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>11,811</td>
<td>11,776</td>
<td>11,391</td>
<td>9501</td>
<td>4811</td>
</tr>
</tbody>
</table>
order $2^7$ group is
\[
\frac{(2^7-1)(2^7-2)(2^7-2^2)}{(2^3-1)(2^3-2)(2^3-2^2)} = \frac{1,984,248}{168} = 11,811.
\]

For an order $2^6$ group the number of subgroups of order $2^3$ is
\[
\frac{(2^6-1)(2^6-2)(2^6-2^2)}{(2^3-1)(2^3-2)(2^3-2^2)} = \frac{234,360}{168} = 1395.
\]
Continuing in a similar fashion the remainder of the left column is filled with the last calculation being
\[
\frac{(2^3-1)(2^3-2)(2^3-2^2)}{(2^3-1)(2^3-2)(2^3-2^2)} = 1.
\]

The second column from the left contains the number of order $2^3$ subgroups that have at least four or more factors. For example if $k=4$ there are $\binom{4}{3} = 4$ subgroups having only three factors. Therefore $15-4=11$ subgroups have four factors. In the same way if $k=5$, then $\binom{5}{3} = 10$ subgroups have only three factors leaving $155-10=145$ subgroups with four or more. Similar calculations finish the column.

For the third column, the number of order $2^3$ subgroups that have at least five or more factors are calculated. For five factors there are $\binom{5}{4} = 5$ different sets of four factors each, and each set has 11 subgroups of order 8. Hence there are $145-5(11)=90$ subgroups of five factors. Similarly, there are $1375-\binom{6}{4} \cdot 11=1210$ subgroups having five or six factors and $11,776-\binom{7}{4} \cdot 11=11,391$ having five, six or seven factors. The fourth and fifth columns are computed in the
same manner.

Therefore 4811 unique $2^3$ subgroups of an order $2^7$ group exist which contain all seven factors. (It is important to note, however, that not all 4811 subgroups are of $R_{\geq III}$.) This number could serve as an upper bound on the number of unique alias subgroups containing all seven factors for $2^{7-3}$ fractions. Similarly 670 is an upper bound for unique alias subgroups of $2^{6-3}$ fractions.

Appendix E contains similar tables for subgroups of order 4, 16, 32, and 64 in addition to repeating Table 2.3.

2.5 SYSTEMS OF LINEAR EQUATIONS

Stoll (1952) emphasizes that the developments of linear algebra are applicable to any linear system for any field. He then provides examples, including fields having sets of remainders modulo $r$, where $r$ is a prime number. Bailey (1959), following Stoll's development, treats the construction of fractional factorials as a linear system over the set of remainders modulo $r$, where $r$ is the number of levels of each factor.

For the case of $r=2$ the specification of a $2^{k-p}$ fraction, $f_1$, as a system of linear equations is
Once $G$ is selected and 0,1 values are assigned to $\{c_i\}$, then

$$f_i = \{x_i \mid x_i \text{satisfies (2.18), and } x_i \in X_k\}.$$ (2.18)

Reducing (2.18) to row echelon form (REF) allows efficient evaluation of the selection of $G$ as demonstrated by Bailey (row echelon form is defined in Appendix A). Let $G_T'$ denote the REF matrix and $G' = (x'_g_1, x'_g_2, \ldots, x'_g_p)$ the set consisting of the row vectors of $G'$. Then (2.19) is an example of $G'$ for $k=7$ and $p=4$.

$$G' = \begin{pmatrix} x'_g_1 \\ x'_g_2 \\ x'_g_3 \\ x'_g_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & a'_{13} & a'_{12} & a'_{11} \\ 0 & 1 & 0 & 0 & a'_{23} & a'_{22} & a'_{21} \\ 0 & 0 & 1 & 0 & a'_{33} & a'_{32} & a'_{31} \\ 0 & 0 & 0 & 1 & a'_{43} & a'_{42} & a'_{41} \end{pmatrix}. \quad (2.19)$$

When $G$ is reduced to $G'$ if any $x'_g_1 = (0, 0, \ldots, 0)$, then $G$ is not an independent set. Secondly if, for all $i$,

$$\|x'_g_i\| \geq 3 \quad (2.20)$$

and if for all $i, j$, where $i$ does not equal $j$,

$$\|x'_g_i + x'_g_j\| \geq 3, \quad (2.21)$$
then \( R > III \) because for any distinct \( i, j, k \)
\[
\| x_i' g_i + x_j' g_j + x_k' g_k \| \geq 3.
\] (2.22)

Lastly (2.18) demonstrates why the problem of selecting feasible fractions cannot be solved directly based upon the empty cells, which are factor level combinations not in the historical data base. The only known quantities are \( E = \{ x_{y_i} \}^* \), the set of empty cells, and \( \overline{E} = \{ x_{z_i} \}^* \), the set of nonempty cells. The problem is to decide for which \( 2^{k-p} \) subset of \( \overline{E} \), if any, there exists a suitable \( G \) such that the \( \{ c_i \} \) can be selected to satisfy (2.18). This can only be solved by an enumeration approach.

2.6 RELATIONSHIPS BETWEEN \( R, k, p \)

2.6.1 Relationships for Wordlengths of the Alias Subgroup

Brownlee, Kelly and Loraine (1948) state four properties regarding the wordlengths of the alias subgroup members being:

1. \[
\sum_{i=1}^{2^p} || x_{a_i} || = k2^{p-1}
\] (2.23)

2. Either \( || x_{a_i} || \) (mod 2) = 0 for all \( i \), or
   \[ || x_{a_i} || \) (mod 2) = 1 for \( 2^{p-1} \) of the \( \{ x_{a_i} \} \).

3. If any \( || x_{a_i} || \) (mod 2) = 1 then the set

---

* Elements of \( E \) are always written with subscripted "y" subscripts and elements of \( \overline{E} \) with subscripted "z" subscripts.
\{x_{a_i} \mid \| x_{a_i} \| (\text{mod} 2) = 0\} \text{ is an order } 2^{p-1} \text{ subgroup, and}

4. If any \( \| x_{a_i} \| = k \), then the remaining elements can be divided into pairs \((x_{a_i}, x_{a_j})\) such that
\[ \| x_{a_i} \| + \| x_{a_j} \| = k. \]

2.6.2 Inequality Relationships for \( R, k, p \)

Using the rationale that with an \( R = III \) fraction all main effects plus an average response can be estimated if higher order interactions are ignored, then at least \( k+1 \) observations must be taken to have the necessary degrees of freedom (Box and Hunter, 1961). Therefore for \( R = III \) it is required that
\[ k + 1 < 2^{k-p}. \tag{2.24} \]

Webb (1968b) and Margolin (1969) proved independently that the minimum number of observations for \( R = IV \) is
\[ 2^{k-p} \geq 2k. \tag{2.25} \]
Combining the results of (2.24) and (2.25) provide an inequality for when the maximum value of \( R = III \):
\[ k + 1 \leq 2^{k-p} \leq 2k. \tag{2.26} \]

Again, using the idea that the number of observations must be greater than or equal to the number of independent estimates, the following inequality for \( R = V \) results (Box and Hunter, 1961)
\[ 1 + k(k + 1)/2 < 2^{k-p}. \tag{2.27} \]
Therefore the maximum value of $R$ is IV when
\[ 2k < 2^{k-p} < 1 + k(k + 1)/2. \quad (2.28) \]

### 2.6.3 Maximum Value of $k$ Given $R$ and $2^{k-p}$

Mitchell (1966) and Draper and Mitchell (1967, 1968) used a computer algorithm to determine the maximum value of $k$ given $R$ and $2^{k-p}$. Their results are shown in Table 2.4.

<table>
<thead>
<tr>
<th>$R$</th>
<th>$2^{k-p}$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>512</td>
<td>23</td>
</tr>
<tr>
<td>V</td>
<td>256</td>
<td>17</td>
</tr>
<tr>
<td>V</td>
<td>128</td>
<td>11</td>
</tr>
<tr>
<td>V</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>V</td>
<td>32</td>
<td>6</td>
</tr>
<tr>
<td>V</td>
<td>16</td>
<td>5</td>
</tr>
<tr>
<td>VI</td>
<td>512</td>
<td>18</td>
</tr>
<tr>
<td>VI</td>
<td>256</td>
<td>12</td>
</tr>
<tr>
<td>VI</td>
<td>128</td>
<td>9</td>
</tr>
</tbody>
</table>

### 2.7 Binary Number Analogy

As discussed in Section 2.1, the search algorithm must perform an exhaustive enumeration of all possible fractions for a given cell of the feasibility matrix. However, what constitutes an exhaustive search is not self-evident. As an approach to the problem, an analogy with the binary number system is used as a beginning point to develop an exhaustive search.
Before discussing the analogy, two items must be defined. A string is defined as a connected sequence of decimal numerals, letters, or binary digits (often referred to as bits) that is treated as a single item (Hughes, 1973, p.729). Concatenation is defined as "the operation that joins two strings in the order specified, thus forming one string whose length is equal to the sum of the lengths of the two strings; it is specified by the operator ||" (Hughes, 1973, p.714).

Since a one-to-one correspondence exists between an alias subgroup, A, say, and a family of fractions, f, it would be most efficient to enumerate A* = {A}, the set of all possible alias subgroups. Unfortunately no direct method to enumerate A* is apparent, although working with the row echelon form generator sets, G', may provide possibilities.

Therefore the necessary but undesirable alternative is to enumerate {G}, the set of all possible generator sets. Since each A has M unique generator sets by equation (2.10), enumerating on {G} entails considerable redundancy. Procedures to minimize this redundancy are developed from the results proven in Section 2.9. The final enumeration scheme is discussed in Chapter 3. The motivation for the basic form of the enumeration is presented next.

In the binary number system when increasing the numbers by "one" from zero (a k-bit string of 0's) to a k-bit string
of 1's, each possible bit string combination occurs once in the sequence. For example if \( k = 3 \), then the sequence is:

\[
\begin{align*}
0 & 0 0 \\
0 & 0 1 \\
0 & 1 0 \\
0 & 1 1 \\
1 & 0 0 \\
1 & 0 1 \\
1 & 1 0 \\
1 & 1 1
\end{align*}
\]

Considering each \( x_{g_1} \) as a bit string and then concatenating \( G \) produces a bit string, \( B \), of length \( k \cdot p \) where

\[
B = x_{g_p} \| x_{g_{p-1}} \| \ldots \| x_{g_2} \| x_{g_1} \| .
\]  

(2.29)

Hence \( B \) is also a binary number. Therefore counting by one from zero to a \((k \cdot p)\)-bit string of 1's enumerates \( \{ G \} \).

However, the above enumeration is inefficient in that:

1) duplicate values of \( x_{g_i} \) are allowed for a given \( G \), and
2) permutations of \( G \) occur. Since \( G \) must be an independent set, a candidate \( G \) cannot have duplicate values. Also, from equation (2.14) it is clear that a permutation of \( G \) generates the same alias subgroup as generated by \( G \). Therefore permutations are redundant. Let \( B_i \) be defined as the corresponding binary number of \( x_{g_i} \). Then to remove from the enumeration all duplication of \( x_{g_i} \) and permutations of \( G \) the following condition is imposed

\[
B_p < B_{p-1} < \ldots < B_2 < B_1.
\]  

(2.30)

To simplify the notation, whenever using condition (2.30) it shall be stated in the form
It is to be understood that ordering on the corresponding binary numbers is intended.

This completes the initial development of the enumeration scheme for \{G\}.

2.8 PROPAGATION OF EMPTY CELLS

Assume for k factors that an "empty cell", denoted $x_{y_i}$, is present in the factorial data base. This means that no experimental unit, $q_i$, in the historical data base, $Q = \{q_i\}$, has for the first k ranked factors a k-tuple of attribute values equivalent to $x_{y_i}$. Given $x_{y_i}$, then what can be said about the occurrence of empty cells when the $(k + 1)^{th}$, $(k + 2)^{th}$, and additional ranked factors are considered?

Assume such an $x_{y_i}$ occurs for k factors. When the search algorithm sorts $Q$ on the first $(k + 1)$ factors, then no $q_j$ can appear with the value $(0, x_{y_i})$, since, if $q_j = (0, x_{y_i})$ for $(k + 1)$ factors, then $q_j = x_{y_i}$ for k factors contradicting the initial assumption of the existence of the empty cell. Similarly, no $q_j$ can equal $(1, x_{y_i})$.

Using the above logic, Table 2.5 depicts the "propagation" of an empty cell, $x_{y_i}$, with the addition of the next two ranked factors.
From the above results two generalizations are made:

1. If the set of empty cells, \( E = \{ x_y \} \), is of order \( e \) for \( k \) factors, then the order of \( E \) is at least \( 2e \) for \( k + 1 \) factors.

2. Given an \( x_y \) for \( k \) factors and defining \( \{ x_{r_j} \} \) as the set of \( 2^r \) possible factorial combinations for the factors \( k + 1, k + 2, \ldots, k + r \), then for \( k + r \) factors there exist \( 2^r \) empty cells, \( \{ x_{y_j} \} \), such that

\[
\{ x_{y_j} \} = \{ x_j \mid x_j = (x_{r_j}, x_y), j = 1, \ldots, 2^r \} . \quad (2.32)
\]
cells. Relationships from each category are then used in the design of the search algorithm as discussed in Chapter III.

2.9.1 Properties of Generator Sets and Alias Subgroups

Some additional notation is needed for the development of the first property. Given a generator set

\[ G = (x_{g_1}, x_{g_2}, ..., x_{g_p}) \]

then \( \overline{G} \) is designated as the \( p \times k \) matrix of \( k \)-tuple row vectors, such that

\[
\overline{G} = \begin{pmatrix}
x_{g_1} \\
x_{g_2} \\
\vdots \\
x_{g_p}
\end{pmatrix}
\]  

(2.33)

The row echelon form* (REF) of \( \overline{G} \) is the \( p \times k \) matrix denoted as \( \overline{G}' \). The row vectors of \( \overline{G}' \), designated \( \{x'_{g_i}\} \), then comprise the REF generator set

\[ G' = (x'_{g_1}, x'_{g_2}, ..., x'_{g_p}) \]  

(2.34)

Uniqueness of Row Echelon Form

Let \( G_1 \) be a generator set for the alias subgroup \( A \) and let \( \overline{G}_1 \) be the \( p \times k \) matrix constructed from \( G_1 \) as specified by equation 2.33. Also let \( \overline{G}' \) be the row echelon form of \( \overline{G}_1 \). The transformation of \( \overline{G}_1 \) to \( \overline{G}' \) consists of elementary

* Row echelon form is defined in Appendix A.
row operations such that

\[ T \overline{G}_1 = \overline{G}' \]  

(2.35)

where \( T \) is the appropriate \( p \times p \) premultiplication matrix [see, for example, Nobel (1977)].

Next let \( G_2 \), not equal to \( G_1 \), be a second generator set of \( A \). Also, again designate \( \overline{G}_2 \) as the \( p \times k \) matrix for \( G_2 \). Since any element of \( G_2 \) can be represented as a linear combination of elements of \( G_1 \), then it must be that there exists a \( p \times p \) premultiplication matrix, \( R \), such that

\[ R \overline{G}_1 = \overline{G}_2 \]  

(2.36)

By symmetry

\[ \overline{G}_1 = R^{-1} \overline{G}_2 \]  

(2.37)

Substituting (2.37) into (2.35), then

\[ T R^{-1} \overline{G}_2 = \overline{G}' \]  

(2.38)

Therefore \( \overline{G}_2 \) has the same \( \text{REF} \) as \( \overline{G}_1 \).

Since the \( \text{REF} \) of a matrix is unique (Nobel, 1977), then \( \overline{G}' \) is the unique \( \text{REF} \) of \( \overline{G}_1 \) and \( \overline{G}_2 \). Hence, because \( G_1 \) and \( G_2 \) can be any generator sets for \( A \), it follows that \( G' \), the generator set consisting of the \( p \) rows \( \overline{G}' \), is the unique \( \text{REF} \) generator set corresponding to \( A \). In summary, for each alias subgroup, \( A \), there corresponds a unique row echelon form generator set, \( G' \).

**Restrictions on Generator Word Lengths**

By the results of the previous section, for every alias subgroup, \( A \), there corresponds a unique \( \text{REF} \) generator set, \( G' \), with corresponding matrix, \( \overline{G}' \). At no loss in generality
it is assumed for the following development that $G'$ can be represented as

$$G' = [I:H]$$  \hspace{1cm} (2.39)

where $I$ is a $p \times p$ identity matrix and $H$ is a $p \times (k-p)$ matrix.

**Theorem 2.1.** Any $2^{k-p}$ fraction, $f_i$, $(p \geq 2)$ can be generated by a generator set $G$, such that

$$3 \leq \|x_{g_i}\| \leq k-p$$

for all $x_{g_i} \in G$.

**Proof:**

1. Since consideration in this work is restricted to fractions for which $R \geq III$, then by definition of resolution, $\|x_{g_i}\| \geq 3$, for all $i$.

2. a. Define

$$q = k - p \hspace{1cm} (2.40)$$

b. Expand (2.39) into a matrix of the form

$$G' = \begin{pmatrix}
  x'_{g_1} \\
  x'_{g_2} \\
  \vdots \\
  x'_{g_p}
\end{pmatrix} = \begin{pmatrix}
  1 & a_{1q} & a_{1q-1} & \cdots & a_{11} \\
  1 & a_{2q} & a_{2q-1} & \cdots & a_{21} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  1 & a_{pq} & a_{pq-1} & \cdots & a_{p1}
\end{pmatrix} \hspace{1cm} (2.41)
$$

c. Let $x'_{g_i}$ have maximum wordlength, i.e.,

$$\|x'_{g_i}\| > \|x'_{g_j}\|, \ j \neq i.$$  

d. If $\|x'_{g_i}\| \leq q$, the theorem is satisfied; so assume $\|x'_{g_i}\| = q + 1$ (it cannot be greater).

e. Therefore $\|x'_{g_j}\| \leq q$ for $j \neq i$ because

i. if $\|x'_{g_j}\| = q + 1$, then
ii. \( \| x'_g + x'_g \| = 2 \), which is a violation of the minimum resolution requirement.

3. Steps 2.d and 2.e imply that \( \| x'_g + x'_g \| \leq q \). Replacing \( x'_g \) by \( (x'_g + x'_g) \), for any \( j \neq i \), results in a generator set \( G \) with the property that \( \| x'_g \| \leq q \) for \( i = 1, 2, \ldots, p \) where:

i. \( x'_g = x'_g \) for \( l \neq i \), and

ii. \( x'_g = x'_g + x'_g \)

The results of Theorem 2.1 are summarized in Figure 2.1, which also includes the \( p = 1 \) case (for which there are no maximum wordlength restrictions).

**Properties Involving Sums of Elements of \( k \)-tuples**

Given

\[ x_i = (a_{ik}, a_{ik-1}, \ldots, a_{i2}, a_{i1}), \quad a_{ij} = 0, 1, \]

where \( x_i \in X_k \) and \( X_k \) is an order \( 2^k \) modulo 2 Abelian group, then \( \{X_i\} \) are all \( 2^k \) possible \( k \)-tuple combinations of \( a_{ij} = 0, 1 \). Therefore there are exactly \( 2^{k-1} \) distinct elements such that

\[ \sum_{i=1}^{2^k} a_{ij} = 2^{k-1}, \quad \text{for any } j. \tag{2.42} \]

Brownlee, Kelly, and Loraine's (1948) restriction on the wordlengths of an alias subgroup (equation 2.23) follows immediately from (2.42). That is for alias subgroup

\[ A = \{x_{q_i}\} : \]
Figure 2.1—Maximum Generator Wordlengths, $\|x_g\|$, Considered for Each Cell of the Feasibility Matrix
1. \[ \sum_{i=1}^{2^p} \| x_{a_i} \| = \sum_{i=1}^{2^p} \sum_{j=1}^{k} a_{ij} \]

2. Reversing the summation, then
\[ \sum_{i=1}^{2^p} \| x_{a_i} \| = \sum_{j=1}^{k} \sum_{i=1}^{2^p} a_{ij} \]

3. Applying (2.42) gives the final result,
\[ \sum_{j=1}^{k} 2^{p-1} = k2^{p-1} \]

or
\[ \sum_{i=1}^{2^p} \| x_{a_i} \| = k2^{p-1} \]

Also, for any fraction \( f_i \), consisting of an order 
\( 2^{k-p} \) set of \( k \)-tuples \( \{x_i\} \), each factor occurs
\( 2^{k-p-1} \) times at the coded "0" level and \( 2^{k-p-1} \) times at the coded "1" level such that
\[ \sum_{i=1}^{2^{k-p}} a_{ij} = 2^{k-p-1} \], for any \( j \) . \hspace{1cm} (2.43)

**Minimum Value of \( k \) Given \( p \)**

Given \( p \), which indicates the degree of fractional replication, it is desired to know what the minimum possible value of \( k \), denoted by \( k_3 \), can be to still have a constructible fraction. From Section 2.6.2 it is clear that the minimum required number of factors increases with resolution, given \( p \) constant. Therefore \( k_3 \) equals the value of \( k \) for which the maximum possible resolution, denoted \( R_{\text{max}} \), is a minimum (i.e., \( R_{\text{max}} = \text{III} \)). Inequality (2.26), which is
\[ (k + 1) \leq 2^{k-p} \leq 2k \]
provides the relationship between k and p when: \( R_{\text{max}} = \text{III} \).

Using (2.26) the values of \( k_3 \) are tabulated in Table 2.6.

<table>
<thead>
<tr>
<th>p</th>
<th>( k_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
</tr>
</tbody>
</table>

**Maximum Generator Values**

In Section 2.7 an analogy with the binary number system is presented as a means of exhaustive enumeration of all possible generator sets, \( \{G\} \), in determining \( A(i,j) \) for some \( i,j \). The question which naturally arises is: Can upper bounds be placed on the binary generator values to avoid redundant enumeration of alias subgroups? To answer this question, some additional characteristics of row echelon form generators are needed.

Row echelon form (REF) is defined in Appendix A. As has been discussed at the beginning of Section 2.9, each alias subgroup has a unique REF generator set, \( G' \). Therefore, the enumeration of all possible alias subgroups would be most efficiently accomplished by enumerating all possible
REF generator sets, \{G'\}. However, such an enumeration has not proven feasible. Therefore, as developed in Section 2.7, an enumeration of all possible generator sets, \{G\}, is employed; but this results in redundant enumeration of alias subgroups. In order to minimize the redundancies, the enumeration of \{G\} "mimics" an enumeration of \{G'\} as much as possible because it is sufficient that every \(G'\) be considered only once. Employing this rationale, the following material develops first the concept of the maximum possible REF generator value (in the binary sense), denoted \(x_{g_{\text{max},i}}\) for the \(i^{\text{th}}\) of \(p\) generators, and secondly uses the \(x_{g_{\text{max},i}}\) values to set upper bounds for \(x_{g_i}\).

To develop the \(x_{g_{\text{max},i}}\) quantities, the following properties of an REF generator set, \(G'\), are needed. \(G'\) consists of a set of \(p\) generators \(x'_{g_1}, x'_{g_2}, \ldots, x'_{g_p}\) where \(x'_{g_i} = (a_{1k}, a_{i,k-1}, \ldots, a_{i2}, a_{i1})\). For each \(G'\):

1. \(x'_{g_1} > x'_{g_2} > \ldots > x'_{g_p}\)
2. \(\|x_{g_i}\| \geq 3\) for all \(i\)
3. If \(a_{iq_i}\), as defined in Appendix A, is the leading element of \(x'_{g_i}\), then \(a_{jq_i} = 0\) for all \(j \neq i\).

From the third property, above, for any \(x'_{g_i}\) there must be a zero element in that position corresponding to the leading element position in each of the other \(p-1\)
generators. Therefore, each $x_{g_i}$ must have at least $p-1$ zero elements; or stated differently, the maximum possible word-length is $k-(p-1)$.

The final property concerns subsets of REF generators

$$x'_{g_i}, x'_{g_{i+1}}, \ldots, x'_{g_p}$$

where $i$ can assume values from 1 to $p$. Suppose $i=p$ such that only the generator $x'_{g_p}$ is being considered. At most $x'_{g_p}$ can have $k-3$ leading zero elements (i.e., $a_{p_k} = a_{p_{k-1}} = \ldots = a_{p_4} = 0$) and still satisfy the word-length restriction. Stated differently, if $a_{p_{q_p}}$ is the leading element for $x_{g_p}$, then $q_p \geq 3$.

The subset of REF generators for $i=p-1$ in (2.44) is $x_{g_{p-1}}, x_{g_p}$. For this subset the minimum value for $q_{p-1}$ is the smallest value such that the minimum resolution requirement is still satisfied. In other words, $x'_{g_{p-1}}, x'_{g_p}$ and $(x'_{g_{p-1}} + x'_{g_p})$ must all have wordlengths of three or greater.

To determine $q_{p-1}$ the results of Table 2.6 are used. Table 2.6 contains the minimum number of factors, $k_3$, which must appear at a nonzero level at least once in a set of generators such that the resolution requirement can be satisfied. Therefore, the minimum value of $q_{p-1}$ must be the value of $k_3$ for two generators as given in Table 2.6. In other words, the leading element for $x'_{g_{p-1}}$ is $a_{p-1,q_{p-1}}$.
where \( q_{p-1} \geq 5 \) (remember that the second subscript for \( a_{i,j} \) increases from right to left in the \( k \)-tuple).

In a similar manner, for the subset consisting of three generators \( x_{g_{p-2}}, x_{g_{p-1}}, x_{g_p} \), the minimum leading element for \( x_{g_{p-2}} \) is, by Table 2.6, \( a_{p-2,6} \). Therefore, it must be that \( q_{p-2} \geq 6 \). It is clear that the minimum leading element position can be specified for the largest binary valued generator of any subset of the form given by (2.44). Table 2.7 summarizes the minimum leading element positions for all possible generator subsets.

<table>
<thead>
<tr>
<th>Subset Size</th>
<th>Minimum Leading Element Position, ( k_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
</tr>
</tbody>
</table>

Now all of the properties of REF generators needed to specify the maximum possible REF generator values, \( \{x'_{g_{\max_1}} \} \), have been stated. It remains to formulate a set of rules, based upon these properties, for constructing an
Rule 1: For \( x'_{g_{\text{max}_i}} \) with leading element \( a_{iq_i} \), then

\[ q_i = k - i + 1. \]

Proof:

1. For an \( \text{REF} \) generator set containing \( x'_{g_{\text{max}_i}} \) it must be that

\[ x'_{g_1} > x'_{g_2} > \ldots > x'_{g_{\text{max}_i}}. \]

2. By definition of row echelon form, for leading elements \( a_{1q_1}, a_{2q_2}, \ldots, a_{iq_i} \) it is required that

\[ q_1 > q_2 > \ldots > q_i. \]

3. a. Therefore, \( q_i \) assumes its largest possible value when

\[ q_1 = k, \]
\[ q_2 = k - 1, \]
\[ q_3 = k - 2, \]
\[ \vdots \]
\[ q_{i-1} = k - (i - 1) + 1. \]

b. Hence, \( q_i = k - i + 1. \)

The final task is to determine the location of the remaining \( k-p \) nonzero elements of \( x'_{g_{\text{max}_i}} \). Conversely, one could determine instead the location of the remaining \( p-i \) zero elements which correspond to leading element positions of generators \( x'_{g_{i+1}}, x'_{g_{i+2}}, \ldots, x'_{g_p} \). Since by definition \( x'_{g_{\text{max}_i}} \) is the largest possible value for the \( i \)th \( \text{REF} \)
generator, then the locations of the $p-i$ remaining zero elements in $x'_{g_{\text{max}_i}}$ must be in the minimum (i.e., the rightmost) possible positions. These minimum possible positions are given by Table 2.7.

Define $z$ to be the number of zero element positions after the leading element yet to be determined for $x'_{g_{\text{max}_i}}$. Therefore, if progressing from left to right in the construction of $x'_{g_{\text{max}_i}}$, then $z$ takes on values $p-i$, $(p-i)-1$, $(p-i)-2$, ..., 2, 1. Now Rule 2 can be stated.

Rule 2: Elements $a_{ij}$ of $x'_{g_{\text{max}_i}}$ for $j<q$ are zero if for a subset size of $z$ [where $z$ takes on values $p-i$, $(p-i)-1$, ..., 2, 1] the value of $k_3$, as given by Table 2.7, equals $j$.

Proof:

1. For $x'_{g_{\text{max}_i}}$ to have its maximum possible value, the $p-i$ zero elements after the leading element must be in the minimum possible positions.
2. Table 2.7 contains the minimum leading element position for $x'_{g_{p-z}}$ where $x'_{g_{p-z}}$ is the largest member of the subset of REF generators $x'_{g_{p-z}}$, $x'_{g_{p-z+1}}$, ..., $x'_{g_p}$.
3. Let $(k_3|z)$ denote the value of $k_3$ for a subset of size $z$. Then it must be, by definition of row echelon form, that $a_{ij} = 0$ if

$$j = (k_3|z), \ z = p-i, p-i-1, \ldots, 2, 1.$$
As an example of the application of Rules 1 and 2, consider the case of \((k, p) = (7, 3)\). For \(x'_{g_{\text{max}_1}}\) the position of the leading element \(a_{1q_1}\) is, using Rule 1,
\[
q_1 = 7 - 1 + 1 = 7
\]
Therefore, the leading element is \(a_{17}\). Using Rule 2 to determine the value of the remaining six elements, then:
1. \(a_{15} = a_{13} = 0\) because \((k_3 \mid 2) = 5\) and \((k_3 \mid 1) = 3\) respectively, and
2. all remaining elements \(a_{16}, a_{14}, a_{12},\) and \(a_{11}\) equal one.

Summarizing the above results,
\[
x'_{g_{\text{max}_1}} = (1101011)
\]

For \(x'_{g_{\text{max}_2}}\) the position of the leading element is
\[
q_2 = 7 - 2 + 1 = 6
\]
By rule 2, \(a_{23} = 0\), such that
\[
a_{25} = a_{24} = a_{22} = a_{21} = 1
\]
Therefore,
\[
x'_{g_{\text{max}_2}} = (0111011)
\]
Finally, for \(x'_{g_{\text{max}_3}}\),
\[
q_3 = 7 - 3 + 1 = 5
\]
Since \(p=3\), there are no zero elements after the leading element, such that
\[
x'_{g_{\text{max}_3}} = (0011111)
\]
Considering each $x'_{\text{gmax}_i}$ as a k-tuple row vector, the above results for $(k,p) = (7,3)$ can be summarized in matrix form as

$$
\begin{pmatrix}
    x'_{\text{gmax}_1} \\
    x'_{\text{gmax}_2} \\
    x'_{\text{gmax}_3}
\end{pmatrix}
= \begin{pmatrix}
    1101011 \\
    0111011 \\
    0011111
\end{pmatrix}
$$

Using Rule 1 and Rule 2, the $x'_{\text{gmax}_i}$ values can be derived for all $(k,p)$ combinations as has been done above for the $(7,3)$ case. Table 2.8 contains the results presented in matrix form.

Now that a table of $x'_{\text{gmax}_i}$ values has been derived, only two more quantities remain to be defined before upper bounds for the binary generator values can be derived. First define $x'_{\text{gmin}_p}$ to be the value that the REF generator $x'_{g_p}$ assumes when one of the remaining REF generators, $x'_{g_{i}}$, equals $x'_{\text{gmax}_i}$. Therefore the position of the leading element for $x'_{\text{gmin}_p}$, as indicated by the subscript $q_p$, is

$$q_p = 3$$

Using the notation:

1. $(l)r$ to represent the repetition of the bit string $r$, $l$ times; and

2. $s|t$ to represent the concatenation of two strings $s$ and $t$,
Table 2.8

Maximum Row Echelon Form Generator Values, \( x'_{\text{max}} \)

<table>
<thead>
<tr>
<th>Number of Factors, ( k )</th>
<th>Value of ( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>10</td>
<td>1010001011</td>
</tr>
<tr>
<td></td>
<td>0110001011</td>
</tr>
<tr>
<td></td>
<td>0011001011</td>
</tr>
<tr>
<td></td>
<td>0001101011</td>
</tr>
<tr>
<td></td>
<td>0000111011</td>
</tr>
<tr>
<td></td>
<td>0000011111</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>110001011</td>
</tr>
<tr>
<td></td>
<td>011001011</td>
</tr>
<tr>
<td></td>
<td>001101011</td>
</tr>
<tr>
<td></td>
<td>000111011</td>
</tr>
<tr>
<td></td>
<td>0000111111</td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11001011</td>
</tr>
<tr>
<td></td>
<td>01101011</td>
</tr>
<tr>
<td></td>
<td>00111011</td>
</tr>
<tr>
<td></td>
<td>000111111</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>1001011</td>
<td>1101011</td>
</tr>
<tr>
<td>0101011</td>
<td>0111011</td>
</tr>
<tr>
<td>0011011</td>
<td>0011111</td>
</tr>
<tr>
<td>0001111</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>101011</td>
<td>111011</td>
</tr>
<tr>
<td>011011</td>
<td>011111</td>
</tr>
<tr>
<td>001111</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>11011</td>
<td></td>
</tr>
<tr>
<td>01111</td>
<td></td>
</tr>
</tbody>
</table>
then \( x'_{g_{\text{min}}_p} \) can be written
\[
x'_{g_{\text{min}}_p} = (k-3)0\mid 111.
\] (2.45)

As an example of a specific value for \( x'_{g_{\text{min}}_p} \), consider the \((k,p) = (7,3)\) case with
\[
x'_{g_1} = x'_{g_{\text{max}}_1}
\]
and
\[
x'_{g_3} = x'_{g_{\text{min}}_3}
\]
Letting \( \bar{G}' \) represent the matrix form of the three generators, then
\[
\begin{pmatrix}
x'_{g_{\text{max}}_1} \\
x'_{g_2} \\
x'_{g_{\text{min}}_3}
\end{pmatrix} =
\begin{pmatrix}
1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & a_{24} & 0 & a_{22} & a_{21} \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{pmatrix}
\]
Lastly, define the \( k \)-tuple \( x_{g_{\text{max}}_i} \) such that
\[
x_{g_{\text{max}}_i} = x'_{g_{\text{max}}_i} + x'_{g_{\text{min}}_p} \quad \text{for } i = 1, \ldots, p-1.
\] (2.46)
The thrust of the following developments are to prove that the above so defined quantities, \( x_{g_{\text{max}}_i} \), are suitable terminating values for the generators, \( x_{g_i} \), in the total enumeration of \( \{G\} \).

To aid in understanding of the following arguments it is important to remember that the inequality
\[
x_{g_p} < x_{g_{p-1}} < \ldots < x_{g_2} < x_{g_1}
\]
is always maintained in the iteration of generator values. In addition for an REF generator set, \( G' \), it is similarly
true that
\[ x'_{g_p} < x'_{g_{p-1}} < \ldots < x'_{g_2} < x'_{g_1}. \]

Therefore in the reduction of a generator set \( G \) to \( G' \), the steps in the reduction can be minimized by a judicious "arrangement" of the \( x_{g_i} \) values. To see this consider the matrix forms \( \bar{G} \) and \( \bar{G}' \) for \( G \) and \( G' \) respectively, where

\[
\bar{G} = \begin{pmatrix}
  x'_{g_1} \\
  x'_{g_2} \\
  \vdots \\
  x'_{g_p}
\end{pmatrix}, \quad \text{and} \quad \bar{G}' = \begin{pmatrix}
  x'_{g_1} \\
  x'_{g_2} \\
  \vdots \\
  x'_{g_p}
\end{pmatrix}.
\]

Because of the inequality relationships, the above matrix arrangement of \( G \) reduces to \( \bar{G}' \) in the fewest number of row operations.

Lemma 2.1. Given \( G \) and \( G' \), the row echelon form of \( G \), then for each \( x'_{g_i} \in G \) and \( x'_{g_i} \in G' \)

\[ x'_{g_i} \leq x_{g_i} \quad \text{for } i=1, \ldots, p. \]

This lemma relies upon the specific form and nature of row echelon form reduction used in the search algorithm. Therefore the proof is not intended to apply to row echelon form reduction in general but only to this specific application. Also, again, the general form of \( x_{g_i} \) is

\[ x_{g_i} = (a_{ik}, a_{ik-1}, \ldots, a_{i2}, a_{i1}) \]

where \( a_{ij} = 0 \) or 1.
Proof:

1. a. In the reduction of G to G', the basic operation is to form a new k-tuple, \( x^*_{g_j} \), say, such that
\[
x^*_{g_j} = x_{g_i} + x_{g_j},
\]
where either
\[
i > j, \text{ or } i < j.
\]
b. Define \( a_{i q_i} \) as the leading nonzero element of \( x_{g_i} \). Then \( x_{g_i} \) is added to \( x_{g_j} \) to form \( x^*_{g_j} \) if \( i < j \) and if \( q_i = q_j \). Therefore since \( a_{i q_i} = a_{j q_j} = 1 \), then
\[
a_{i q_i} + a_{i q_j} = 0
\]
(by the rule of composition for an additive Abelian group) such that
\[
x^*_{g_j} < x_{g_j}.
\]
2. \( x_{g_i} \) is combined with \( x_{g_j} \), for \( i > j \), to form a new k-tuple, \( x^*_{g_j} \), if \( a_{j q_i} = 1 \) (i.e., \( x_{g_i} \) is added to \( x_{g_j} \) to remove the nonzero element from the \( q_i^{th} \) position in \( x_{g_j} \)).
Then
\[
a^*_{j q_i} = a_{j q_i} + a_{i q_i}
\]
\[
= 1 + 1 = 0
\]
such that
\[ x^*_g_j < x_{g_j} \]
(e.g., let \( x_{g_1} = 1 0 1 0 0 1 1 \) and \( x_{g_3} = 0 0 1 1 1 1 0 \), then
\[ x_{g_1} + x_{g_3} = x^*_g_1 = 1 0 0 1 1 0 1 < x_{g_1} \).

3. a. The only other operation that occurs in the reduction of \( G \) to \( G' \) is that operation equivalent to an interchange of rows when working in matrix form. This can only occur when two intermediate values in the reduction process, say \( x^*_i \) and \( x^*_j \) (\( i < j \)) have leading elements \( a^*_{iq_i} \) and \( a^*_{jq_j} \) such that
\[ q_j > q_i \]
(e.g., \( x^*_2 = 0 0 1 1 0 1 1 \)
\[ x^*_3 = 0 1 0 1 1 0 0 \],
such that \( q_2 = 5, q_3 = 6 \).

b. However based upon the results of steps 1 and 2 and that:
   i. \( x_{g_p} < \ldots < x_{g_2} < x_{g_1} \),
   ii. \( x^*_j \leq x_{g_j} < x_{g_i} \), and
   iii. \( x^*_i < x^*_j \)
then it must follow that
\[ x^*_j < x_{g_i} \], and
\[ x^*_i < x_{g_j} \].
4. Therefore from steps 1, 2, and 3 it follows that
\[ x'_{g_i} \leq x_{g_i}, \text{ for } i = 1, \ldots, p. \]

Lemma 2.2. Given \( G \) with corresponding \( G' \) such that for every \( x'_{g_i} \in G' \), \( \| x'_{g_i} \| \leq k-p \). Then in the iteration of generator sets, \( G' \) is considered before \( G \) or else \( G' = G \).

Proof:
1. If \( x_{g_i} \in G \) and \( x'_{g_i} \in G' \), then by Lemma 2.1
   \[ x'_{g_i} \leq x_{g_i} \text{ for } i = 1, \ldots, p. \]

2. If for at least one \( i \), \( x'_{g_i} < x_{g_i} \), then by the specification of the generator iteration, \( G' \) is considered before \( G \).

3. Otherwise \( G' = G \).

Lemma 2.3. Given \( G \) for which \( \| x_{g_i} \| \leq k-p \) for \( i = 1, \ldots, p \) with corresponding \( G' \) such that for at most one \( x'_{g_j} \in G' \), \( \| x'_{g_j} \| = k-p+1 \) where \( j < p \). Then \( G' \) can be transformed to \( G* \) where \( x^*_{g_i} \in G* \) such that:
   i. \( \| x^*_{g_i} \| \leq k-p, \text{ for } i = 1, \ldots, p, \text{ and} \)
   ii. \( x^*_{g_i} \leq x_{g_i} \).

Proof:
1. By Lemma 2.1, \( x'_{g_i} \leq x_{g_i}, \text{ for } i = 1, \ldots, p. \)

2. a. Let \( x'_{g_j}, j < p, \) be the one \( k \)-tuple for which
   \[ \| x'_{g_j} \| = k-p+1. \]
b. Since \( \| x_{g_i} \| \leq k-p \) for any \( i \), it must be that

\[
x'_{g_j} < x_{g_j}
\]

because they cannot be equal if they have different wordlengths.

3. As demonstrated in Theorem 2.1,

\[ \| x'_{g_j} + x'_{g_i} \| \leq k-p \] for any \( i \neq j \).

4. a. Let \( A \) be the alias subgroup of \( G \). For \( j < p \), since 

\( x'_{g_p} \) has the smallest binary value in \( A \), then define 

\( x^*_{g_j} \) such that

\[
x^*_{g_j} = x'_{g_j} + x'_{g_p}
\]

is the smallest member of \( A \) with the properties that 

\( x^*_{g_j} > x'_{g_j} \) and \( \| x^*_{g_j} \| \leq k-p \).

b. Because \( x_{g_j} \in A \) and \( x_{g_j} > x'_{g_j} \), then it must be that 

\[
x_{g_j} \geq x^*_{g_j}
\]

thus completing the proof.

The following theorem, Theorem 2.2, proves that the 

\( x_{g_{\text{max}_i}} \) values for \( i < p \), as defined by equation 2.46, can be 

used as upper bounds in the iteration of \( x_{g_i} \) values. The 

quantities, \( x'_{g_{\text{max}_i}} \), referred to in the proof, are the max­

imum REF values as given in Table 2.8.

Theorem 2.2. Given \( G \) with alias subgroup \( A \) and with one or more \( x_{g_i} \) such that \( x_{g_i} > x_{g_{\text{max}_i}} \), \( i < p \). Then
A has been previously considered in the enumeration of generator sets. (Note: the $p^{\text{th}}$ bound, $x_{\text{gmax}_p}$, is not included in this proof.)

Proof:

1. a. Let $G' = \{ x_1' \}$ be the REF of $G$. As a result of Lemma 2.2 or Lemma 2.3, whichever is applicable, the alias subgroup $A$ has been previously considered if it can be shown for at least one $x_{g_i} \in G$ that $x_{g_i}' < x_{g_i}$.

   b. It is given that $x_{g_j} > x_{\text{gmax}_j}$ for one or more $j$, where $j < p$. Considering just one generator, say the $i^{th}$ one for which $x_{g_i} > x_{\text{gmax}_i}$, then if it can be shown that $x_{g_i}' < x_{g_i}$, this is sufficient to prove that $A$ has been previously considered.

2. a. Two cases exist. Either $\| x_{g_i}' \| \leq k-p$ or $\| x_{g_i}' \| = k-p+1$.

   b. Let $\| x_{g_i}' \| \leq k-p$. Since by definition, a property of $x_{g_i}'$ is $\| x_{\text{gmax}_i}' \| = k-p+1$, then $x_{g_i}' < x_{\text{gmax}_i}'$ because they cannot be equal if they have different
c. However it is given that

\[ x_{\text{gmax}_i} < x_{g_i} \]

and by definition of \( x_{\text{gmax}_i} \) (equation 2.46)

\[ x'_{\text{gmax}_i} < x_{\text{gmax}_i} \]

Therefore

\[ x'_{g_i} < x_{g_i} \]

which is sufficient to prove that A has been previously considered.

3. a. For the second case let \( || x'_{g_i} || = k-p+1 \). By Lemma 2.3, \( x'_{g_i} \) can be transformed to \( x*_{g_i} \) where

\[ x*_{g_i} = x'_{g_i} + x'_{g_p} \quad \text{for } i < p \]

such that

\[ || x*_{g_i} || \leq k-p \]

and

\[ x*_{g_i} \leq x_{g_i} \]

b. It is now necessary to prove that

\[ x*_{g_i} \leq x_{\text{gmax}_i} \]

i. By equation 2.46,

\[ x_{\text{gmax}_i} = x'_{\text{gmax}_i} + x'_{\text{gmin}_p} \]

and as stated above

\[ x*_{g_i} = x'_{g_i} + x'_{g_p} \]

ii. Define \( j_{\text{max}_r} \) as the position of the \( r \)th zero element, counting from right to left,
of \( x'_{g_{\text{max}_i}} \). Therefore by definition of

\[ x'_{g_{\text{max}_i}} \]

\[
\begin{align*}
\text{jmax}_1 &= 3 \\
\text{jmax}_2 &= 5 \\
\text{jmax}_3 &= 6 \\
\end{align*}
\]

and so forth for the \( p-1 \) zero elements.

iii. Similarly define \( j_r \) for \( x'_{g_i} \).

iv. Since the zero elements of \( x'_{g_{\text{max}_i}} \) occupy the

lowest binary positions by definition, then

\[ j_r > \text{jmax}_r \quad \text{for all } r \]

v. Because the zero element positions of \( x'_{g_{\text{max}_i}} \)
correspond to leading element positions of

\[ x'_{g_{i+1}}, \ldots, x'_{g_{\text{min}_p}} \],

then

\[ x'_{g_{\text{max}_i}} = x'_{g_{\text{max}_i}} + x'_{g_{\text{min}_p}} \]

has a nonzero element in the \((\text{jmax}_i)\)th position. Similarly, \( x^*_{g_i} \) has a nonzero element

in the \((j_1)\)th position.

vi. If

\[ j_r > \text{jmax}_r \]

for at least one \( r > 1 \), then

\[ x'_{g_i} < x'_{g_{\text{max}_i}} \]

and it follows that

\[ x^*_{g_i} < x_{g_{\text{max}_i}} \].
vii. If \( j_r = j_{\text{max}_r} \) for all \( r > 1 \) but
\[ j_1 > j_{\text{max}_1}, \]
then it must be that
\[ x'_g = x || 0111 \]
and
\[ x'_{\text{gmax}_1} = x || 1011 \]
where \( x \) is defined as the leading \( k-4 \) elements of \( x'_g \), which equal, because of the stated conditions, the \( k-4 \) leading elements of \( x_{\text{gmax}_1} \).

viii. For this case it must also be that
\[ x'_g_p = (k-4)0 || 1011 \]
and
\[ x'_{\text{gmin}_p} = (k-4)0 || 0111 \]

ix. Therefore
\[ x^*_g = x_{\text{gmax}_1} = x || 1100 \]

x. Lastly, if \( j_r = j_{\text{max}_r} \) for all \( r \), then
\[ x^*_g = x_{\text{gmax}_1} \]

xi. Therefore for all possible conditions
\[ x^*_g \leq x_{\text{gmax}_1} \]

c. However it is given that
\[ x_{\text{gmax}_1} < x_g \]
Therefore
\[ x^*_g < x_g \]
which is sufficient to prove that \( A \) has been previously considered, thus completing the proof.
Therefore by the results of the above theorem, the values \( x_{g_{\text{max}i}} \), for \( i = 1, \ldots, p-1 \), can be used as the terminating values in the iteration of generators, \( x_{g_i} \), without excluding any alias subgroup from consideration.

For the \( p^{\text{th}} \) generator, \( x_{g_p} \), a conservative terminating value, \( x_{g_{\text{max}p-1}} \), is selected. While this terminating value results in redundant consideration of alias subgroups, other checks on the \((x_{g_p}, x_{g_{p-1}})\) pair are possible, as discussed in Section 3.3.2, thereby minimizing the redundancy problem.

Table 2.9 contains the terminating values, \( x_{g_{\text{max}i}} \) for generators, \( x_{g_i} \), for all combinations of \( k \) and \( p \).

It is useful for the search algorithm to be able to calculate \( x_{g_{\text{max}i}} \) for \( i = 1, \ldots, p \) given \( k, p \). However before stating the formulas, it is helpful to review some notation.

Let \((m)_r\) represent the repetition of the bit string \( r \), \( m \) times. For example

\[
\begin{align*}
(4) 0 & = 0 0 0 0 0 \\
(2.47)
\end{align*}
\]

Then the concatenation of two strings, say \( s \) and \( t \), where

\[
\begin{align*}
s & = (4) 0, \quad \text{and} \\
t & = (5) 1,
\end{align*}
\]

can be written

\[
\begin{align*}
s \| t & = (4) 0 \| (5) 1 \\
& = 0 0 0 0 1 1 1 1 1
\end{align*}
\]
### Table 2.9

**Generator Terminating Values**, $x_{\text{gmax}_1}$

<table>
<thead>
<tr>
<th>Number of Factors, $k$</th>
<th>Value of $p$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1010001100</td>
</tr>
<tr>
<td></td>
<td>0110001100</td>
</tr>
<tr>
<td></td>
<td>0011001100</td>
</tr>
<tr>
<td></td>
<td>0001101100</td>
</tr>
<tr>
<td></td>
<td>0000111100</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>110001100</td>
</tr>
<tr>
<td></td>
<td>011001100</td>
</tr>
<tr>
<td></td>
<td>001101100</td>
</tr>
<tr>
<td></td>
<td>000111100</td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11001100</td>
</tr>
<tr>
<td></td>
<td>01101100</td>
</tr>
<tr>
<td></td>
<td>00111100</td>
</tr>
<tr>
<td></td>
<td>00111100</td>
</tr>
</tbody>
</table>
Table 2.9 (continued)

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>1001100</td>
<td>1101100</td>
<td>1111100</td>
</tr>
<tr>
<td></td>
<td>0101100</td>
<td>0111100</td>
<td>1111100</td>
</tr>
<tr>
<td></td>
<td>0011100</td>
<td>0111100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0011100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>101100</td>
<td>111100</td>
</tr>
<tr>
<td></td>
<td>011100</td>
<td>111100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>11100</td>
<td>11100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* The $x_{gmax_i}$ values are arranged in the order $x_{gmax_1}$, $x_{gmax_2}$, ..., $x_{gmax_p}$. 
(Note: string and concatenation are defined in Section 2.7).

For the $i^{th}$ generator, define the quantity $h$ to be

$$h = p - i, \quad i = 1, 2, \ldots, p - 1.$$  

Secondly, define $H(h)$ and $w(h)$ to have the values as given in Table 2.10. Then $x_{g\max_i}$, for $i < p$, is given by the formula

$$x_{g\max_i} = (i-1) 0 || [k-(i-1)-w(h)] 1 || H(h)$$

$$i=1, \ldots, p-1 \quad (2.48)$$

<table>
<thead>
<tr>
<th>$h$</th>
<th>$H(h)$</th>
<th>$w(h)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1100</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>01100</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>001100</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>0001100</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>010001100</td>
<td>9</td>
</tr>
</tbody>
</table>

2.9.2 Feasibility Matrix Cell Dependencies

Three types of dependencies are considered between cells of the feasibility matrix:

1. dependencies of cells in a common column,
2. dependencies within a row, and
3. dependencies for the p=1 diagonal.

The region of concern is the fractional factorial region of the feasibility matrix.
A cell of the feasibility matrix is denoted by $A(k, p)$. If at least one fraction, $f_j$, of $2^{k-p}$ observations is feasible, then $A(k, p) = 1$. Otherwise, $A(k, p) = 0$.

**Column Rule**

Theorem 2.3. (Column Rule) If $A(k, p) = 0$, then $A(k+1, p+1) = 0$.

Proof:

The approach is to show, equivalently, that $A(k+1, p+1) = 1$ implies $A(k, p) = 1$.

1. Assume $A(k+1, p+1) = 1$, such that there exists a fraction $f_j$ satisfying:

\[ f_j = \{ x_i \mid \mathbf{G} \cdot x_i = C, \, i=1, \ldots, 2^{(k+1)-(p+1)} \} \]

where:

i. $\mathbf{G}$ is the $(p+1) \times (k+1)$ matrix form of the generator set,

ii. $G = (x_{g_1}, x_{g_2}, \ldots, x_{g_{p+1}})$ for $f_j$, and

iii. $C$ is a $(p+1) \times 1$ binary column vector.

2. a. The alias subgroup is

\[ A = \{ x_{a_i} \mid x_{a_i} = \sum_{j=1}^{p+1} A_j x_{g_j}, \text{ for all } A_j = 0 \text{ or } 1 \} \]

where $x_{a_i}$ is a binary $(k+1)$-tuple of the form

\[ x_{a_i} = (a_{a_1 k+1}, a_{a_1 k}, \ldots, a_{a_1 2}, a_{a_1 1}) \]

b. Consider two exhaustive cases.

case 1: $a_{a_i k+1} = 1$ for $2^{(p+1)-1}$ of the $x_{a_i} \in A$ by equation 2.42, or
case 2: \( a_{a_i,k+1} = 0 \) for all \( x_{a_i} \in A \) (case 2 can occur when \( f_j \) has resolution less than the maximum possible resolution given \( k, p \)).

3. Starting with case 1, define

\[
A_v = \{ x_{v_i} \mid x_{v_i} \cdot (1, 0, \ldots, 0) = 0, \text{ and } x_{v_i} \in A \}
\]

where \( x_{v_i} \cdot (1, 0, \ldots, 0) \) is the inner product modulo 2 as defined in equation 2.12.

4. Therefore \( A_v \) is an order \( 2^p \) subgroup of \( A \), and all \( x_{v_i} \) are of the form \((0,x^*_{v_i})\), such that

\[
x^*_{v_i} = (a_{v_i,k}, a_{v_i,k-1}, \ldots, a_{v_i,2}, a_{v_i,1})
\]

5. Let \( G_v \) be a \( px(k+1) \) matrix whose rows are a generator set, \( G_y \), for \( A_v \). Then \( G_v \) is of the form

\[
G_v = [0:G^*_v]
\]

where \( G^*_v \) is a \( pxk \) matrix.

6. Treating

\[
[0:G^*_v] \cdot x_i = C^*
\]

as a \( k+1 \) column vector, then for all \( x_i \in f_j \)

\[
[0:G^*_v] \cdot x_i = C^* = G^*_v \cdot x^*_{v_i} = C^*
\]

which is equivalent to \( A(k, p) = 1 \).

7. For case 2, since \( a_{a_i,k+1} = 0 \) for all \( x_{a_i} \in A \), then if \( G \)
is a generator set for $A$, each generator $x_{g_i}$ is of the form

$$x_{g_i} = (0, a_{g_i k}, a_{g_i k-1}, \ldots, a_{g_i 1})$$

Define

$$x^*_{g_i} = (a_{g_i k}, a_{g_i k-1}, \ldots, a_{g_i 1})$$

such that

$$x_{g_i} = (0, x^*_{g_i})$$

9. Letting the matrix form of $G$ be $\overline{G}$, then

$$\overline{G} = [0: \overline{G}^*]$$

such that for each $x_i \in f_j$

$$\overline{G} \cdot x_i = C$$

$$= [0: \overline{G}^*] \cdot x^*_{i}$$

$$= \overline{G}^* \cdot x^*_{i} = C$$

which is again equivalent to $A(k, p) = 1$. This completes the proof.

**Generator Subset Rule**

A corollary to the Column Rule concerns the case of individual generator sets, $G$.

**Corollary 2.3.1.** Given an independent generator set $G^* = (x^*_{g_1}, x^*_{g_2}, \ldots, x^*_{g_p})$ and a second generator set $G = (x_{g_1}, x_{g_2}, \ldots, x_{g_p}, x_{g_{p+1}})$, such that

$$x_{g_i} = (0, x^*_{g_i}) \text{ for } i = 1, \ldots, p$$

$x_{g_{p+1}}$ is arbitrarily specified except that it is independent of the other
members of $G$). Then if
\[ [A(k,p) \mid G^*] = 0 \]
it follows that
\[ [A(k+1, p+1) \mid G] = 0 \]

It is shown, equivalently, that if
\[ [A(k+1, p+1) \mid G] = 1, \text{ then } [A(k, p) \mid G^*] = 1. \]

Proof:

1. Assume \([A(k+1, p+1) \mid G] = 1\) such that the fraction, $f_j$, with $2(k+1)-(p+1)$ observations \(\{x_i\}\), is feasible.

Then there exist $c_j$ such that
\[
(0, x^*_{g_1}) \cdot x_i = c_1 \\
(0, x^*_{g_2}) \cdot x_i = c_2 \\
\vdots \\
(0, x^*_{g_p}) \cdot x_i = c_p
\]
for all $x_i \in f_j$.

2. Define $x^*_{i}$ such that
\[
x_i = (a_i, k+1, x^*_{i}).
\]

It follows that
\[
x^*_{g_1} \cdot x^*_{i} = c_1 \\
x^*_{g_2} \cdot x^*_{i} = c_2 \\
\vdots \\
x^*_{g_p} \cdot x^*_{i} = c_p
\]
for all $x_i \in f_j$. Therefore $A(k, p) = 1$ thus completing the proof.
Diagonal Rule for p = 1 Case

Theorem 2.4. (Diagonal Rule) If \(A(k, 1) = 0\), then \(A(k+1, 1) = 0\).

The approach is to show equivalently that \(A(k+1, 1) = 1\) implies \(A(k, 1) = 1\).

Proof:

1. Assume \(A(k+1, 1) = 1\) such that the fraction, \(f_j\), with \(2^{k+1} - 1\) observations \(\{x_i\}\), is feasible. Then for either \(c_1 = 0\) or \(c_1 = 1\) there exists a generator \(x_{g_1}\) such that

\[x_{g_1} \cdot x_i = c_1\]

for all \(x_i \in f_j\).

2. a. Define \(x^*_{g_1}\) such that

\[x_{g_1} = (a_{g_1, k+1}, x^*_{g_1})\]

b. Consider the order \(2^{k-1}\) subset of \(\{x_i\}\) for which \(a_{i, k+1} = 0\). Also define \(x^*_i\) for this subset such that \(x_i = (0, x^*_i)\).

c. Then

i. \(x_{g_1} \cdot x_i = c_1\),

which can be rewritten as

ii. \((a_{g_1, k+1}, x^*_{g_1}) \cdot (0, x^*_i) = c_1\), or

iii. \(x^*_{g_1} \cdot x^*_i = c_1\) for \(i = 1, \ldots, 2^{k-1}\).

d. Therefore \(A(k, 1) = 1\) thus completing the proof.
Designate the first ranked factor for which a full factorial is not feasible as $k_{\min}$. Then define, for $p = 1$, and $k > k_{\min}$,

$$x_{g_{\max}} = (k-k_{\min}) 0 || (k_{\min}) 1$$

(2.48)

(using the notation demonstrated by equation (2.47), again for example, $(4)0 = 0 0 0 0$).

Corollary 2.4.1. Given $p = 1$, $k > k_{\min}$, $x_{g_{\max}}$ as defined in (2.48) and a candidate generator $x_{g_1}$ such that (using the binary number analogy discussed in Section 2.7)

$$x_{g_1} > x_{g_{\max}}$$

then $[A(k, 1) \mid x_{g_1}] = 0$.

This result is proven by induction.

Proof:

1. To simplify the notation let $u = k_{\min}$. Also define $x_{y_1}$ to be an empty cell in the first $k_{\min}$ ranked factors (one exists, by definition of $k_{\min}$).

2. For the $(u + 1)$-tuples in the factorial data base there must be, based upon the results of Section 2.8, two empty cells of the form

$$x'_{y_1} = (0, x_{y_1})$$

and

$$x^*_{y_1} = (1, x_{y_1})$$

3. Consider the candidate generator

$$x_{g_1} = (1, a_{g_1}u, a_{g_1}u-1, \ldots, a_{g_1}2, a_{g_1}1)$$

Two possibilities exist. Either
4. For either possibility, it must be that
\[ [A(k_{\text{min}} + 1, 1) | x_{g_1}] = 0. \]

5. a. Now consider some arbitrary \( k > k_{\text{min}} \) such that for
the \( k \)-tuple generator \( x_{g_1} \),
\[ x_{g_1} > x_{\text{gmax}}. \]

b. At no loss in generality assume for \( x_{g_1} \) that
\( a_{g_1 k} = 1 \) such that \( x_{g_1} \) can be written
\[ x_{g_1} = (1, x) \]
where
\[ x = (a_{g_1 k-1}, \ldots, a_{g_1 2}, a_{g_1 1}). \]

6. Given the empty cell, \( x_{y_i} \), in \( k_{\text{min}} \) factors, then from
the results of Section 2.8, for \( k \) factors there must be
two empty cells of the form
\[ x'_{y_i} = (0, x_{y}) \]
and
\[ x^*_{y_i} = (1, x_{y}), \]
where \( x_{y} \) is a \( k-1 \) tuple such that
\[ x_{y} = (u, x_{y_i}) \]
and \( u \) is any arbitrary \((k-k_{\text{min}}-1)\)-tuple.

7. Then one of two possibilities exist. Either
\[ x_{g_1} \cdot x'_{y_i} = 0, \quad x_{g_1} \cdot x^*_{y_i} = 1; \]
or
\[ x_{g_1} \cdot x' y_i = 1 \quad \text{and} \quad x_{g_1} \cdot x^* y_i = 0 \].

8. For either possibility it must be that
\[ [A(k, 1) \mid x_{g_1}] = 0 \]
thus completing the proof.

Corollary 2.4.2. If \([A(k, 1) \mid x_{g_1}] = 0\), then
\[ [A(k+1, 1) \mid (0, x_{g_1})] = 0. \]

The approach is to show equivalently that
\([A(k+1, 1) \mid (0, x_{g_1})] = 1 \implies [A(k, 1) \mid x_{g_1}] = 1. \]

Proof:
1. Assume \([A(k+1, 1) \mid (0, x_{g_1})] = 1\) such that the fraction, \(f_j\), with \(2^{(k+1)-1}\) observations \(\{x_i\}\), is feasible. Then for either \(c_1 = 0\) or \(c_1 = 1\) there exists a generator \((0, x_{g_1})\) such that
\[(0, x_{g_1}) \cdot x_i = c_1 \]
for all \(x_i \in f_j\).
2. a. Define \(x^*_{i} \) such that
\[ x^*_i = \begin{pmatrix} a_{i} \cdot k+1 \\ x^*_i \end{pmatrix} \cdot x_i \]
b. Then
i. \((0, x_{g_1}) \cdot x_i = c_1\), or
ii. \((0, x_{g_1}) \cdot \begin{pmatrix} a_{i} \cdot k+1 \\ x^*_i \end{pmatrix} = x_{g_1} \cdot x^*_i = c_1 \)
for all \(x_i \in f_j\). Therefore \([A(k, 1) \mid x_{g_1}] = 1\) thus
completing the proof.

Row Cell Dependencies

Once a cell of the feasibility matrix has been determined to be feasible such that \( A(k, p) = 1 \), what can be concluded regarding \( A(k, p') \), where \( p' > p \)? The condition \( A(k, p) = 1 \) requires the existence of a fraction \( f_j \), consisting of \( 2^{k-p} \) nonempty cells, \( \{x_{z_j} \} \), and a generator set, \( G \), for \( f_j \). If a \((p+1)\)th independent generator, \( x_{g_{p+1}} \), can be selected such that the generator set

\[
G^* = (x_{g_{p+1}}, G)
\]

generates a fraction \( f^*_j \), consisting of the set of nonempty cells

\[
\{x^*_z \mid x^*_z \in \{x_{z_j} \}, j = 1, \ldots, 2^{k-p-1} \}
\]

and if the minimum resolution requirement (i.e., \( R \geq III \)) is satisfied, then \( A(k, p+1) = 1 \) since the set corresponding to \( f_j \) is nonempty. Therefore the existence of a satisfactory generator, \( x_{g_{p+1}} \), is a sufficient condition for

\( A(k, p+1) = 1 \), given \( A(k, p) = 1 \). This sufficiency argument is next extended to larger values of \( p \), say \( p' \), for which the following holds (equation 2.24):

\[
k + 1 \leq 2^{k-p'}
\]

Theorem 2.5. (Row Rule) Given \( 8 \leq k \leq 9 \) and \( A(k, p) = 1 \), then \( A(k, p') = 1 \), for all \( p' > p \) with the restriction that \( k + 1 \leq 2^{k-p'} \). For \( k = 7 \), \( p < 3 \), and \( A(7, p) = 1 \) then \( A(7, p') = 1 \) for \( p' = p + 1, \ldots, 3 \). For \( k = 7 \), \( p \leq 3 \), and
A(7, p) = 1 for some generator set G with alias subgroup A, then A(7, 4) = 1 if \[ \| x_{a_1} \| \neq 5 \text{ or } 6 \] for any \( x_{a_1} \in A \).

The proof is to demonstrate by total enumeration that \( x_{g_{p+1}} \) can always be selected which satisfies the \( R \geq III \) requirement.

Proof:

Case A, \( k = 7 \):

1. First consider \( k = 7 \) and \( p < 3 \). For \( p = 0 \) or \( p = 1 \), a first or second generator, respectively, can obviously be selected.

2. a. For \( p = 2 \) let \( G = (x_{g_1}, x_{g_2}) \) be a generator set for which \( A(7, 2) = 1 \). Also let \( G' = (x'_{g_1}, x'_{g_2}) \) be the row echelon form (REF) of \( G \) and \( \overline{G}' \) the matrix with \( x'_{g_1} \) and \( x'_{g_2} \) as row vectors.

b. At no loss in generality \( \overline{G}' \) has the form

\[
\overline{G}' = \begin{pmatrix}
    x'_{g_1} \\
    x'_{g_2}
\end{pmatrix} = \begin{pmatrix}
    1 & 0 & a_1 & a_{14} & a_{13} & a_{12} & a_{11} \\
    0 & 1 & a_{25} & a_{24} & a_{23} & a_{22} & a_{21}
\end{pmatrix}
\]

c. From Table 2.6, for \( p = 2 \), the minimum number of factors (\( k_3 \)) that must appear at a nonzero level at least once (i.e., for \( R \geq III \)) in either \( x'_{g_1} \) or \( x'_{g_2} \) is five. Therefore at most two columns of \( \overline{G}' \) can be all zeros.

d. If at least one column of \( \overline{G}' \) is all zeros, say the fifth column, then a third generator \( x_{g_3} \) can be
selected of the form
\[ x_{g_3} = (0 \ 0 \ 1 \ a_{34} \ a_{33} \ a_{32} \ a_{31}) , \]
where the values of \( a_{34}, a_{33}, a_{32} \) and \( a_{31} \) are chosen to provide an \( R \geq III \) fraction. For example if
\[ \overline{G}' = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} , \]
then
\[ x_{g_3} = (0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1) \]
is a suitable third generator.

e. Table 2.11 enumerates all possible wordlength combinations and a suitable \( x_{g_3} \) for when no column of \( G' \) has all zeros. Only distinct forms of \( G' \) are listed. Two \( G' \) matrices are distinct if one form cannot be obtained from the other by an interchange of columns.

3. a. It remains for the \( k = 7 \) case to prove that
\( A(7, 4) = 1 \) when \( A(7, p) = 1 \) \((p<3)\) for some generator set \( G \) with alias subgroup \( A \) such that for any \( x_{a_i} \in A, \| x_{a_i} \| \neq 5 \) or 6. For \( p = 1 \) there are three possibilities, either \( \| x_{a_i} \| = 3, 4 \) or 7 (note: for \( p = 1 \) the alias subgroup consists only of \( x_{g_1} \) and the identity element).

b. At no loss in generality for \( \| x_{a_i} \| = 3 \) let
\[ x_{g_1} = 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 . \]
Table 2.11

Remaining Wordlength Combinations for Distinct $G'$ and a Satisfactory $x_g3$
for $(k, p) = (7, 2)$

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x'g_1$</th>
<th>$x'g_2$</th>
<th>$\overline{G}'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td></td>
<td>1 0 1 1 1 0 0 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 0 0 0 1 1 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
<td>1 0 1 1 1 0 0 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 0 0 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 1 1 1 0 1</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td></td>
<td>1 0 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 0 0 0 1 1 1</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 1 1 0 1 1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td></td>
<td>1 0 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 0 0 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 1 1 0 1 1</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
<td>1 0 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 0 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 1 1 0 0 1 1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td></td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 1 1 0 0 0 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td></td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 1 1 1 0 0 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 0 1 1 1 1</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td></td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>x_g3</td>
<td></td>
<td>0 0 0 0 1 1 1 1</td>
</tr>
</tbody>
</table>
Then three suitable additional generators are
\[ x_{g_2} = 0100101 \]
\[ x_{g_3} = 0010011 \]
\[ x_{g_4} = 0001111 \]
such that \( A(7, 4) = 1 \).

c. For \( ||x_a|| = 4 \) let
\[ x_{g_1} = 1000111 \]
Then three suitable additional generators are
\[ x_{g_2} = 0100101 \]
\[ x_{g_3} = 0010011 \]
\[ x_{g_4} = 0001110 \]
d. For \( ||x_a|| = 7 \) then \( x_{g_1} = 1111111 \). Three additional generators are
\[ x_{g_2} = 1000111 \]
\[ x_{g_3} = 0100110 \]
\[ x_{g_4} = 0010011 \]
Therefore \( A(7, 4) = 1 \) for each of the three possible cases for \( x_{g_1} \).

4. For \( p = 2 \), Table 2.12 contains all distinct \( x'_{g_1}, x'_{g_2} \) pairs, which satisfy the wordlength restrictions for \( x_a \), plus two suitable additional generators to prove that \( A(7, 4) = 1 \).
Table 2.12

For (k, p) = (7, 2) All Distinct \( \bar{G}' \) Which Satisfy \( \| x_{a_i} \| \neq 5 \) or 6 and a Satisfactory \( x_{g_3} \) and \( x_{g_4} \)

Wordlengths

<table>
<thead>
<tr>
<th>( x'g_1 )</th>
<th>( x'g_2 )</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>1000110 0100101 0010011 0001111</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>1000111 0100110 0010101 0001011</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>1000111 0101110 0010110 0001011</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>1000111 0101110 0010110 0001011</td>
</tr>
</tbody>
</table>
5. In a similar manner Table 2.13 contains, for \( p = 3 \), all distinct \( x_1^p, x_2^p, x_3^p \) combinations which satisfy the wordlength restrictions for \( x_a \) plus a suitable \( x_{g_4} \) value to prove that \( A(7, 4) = 1 \).

Case B, \( k = 8 \):

6. For \( k = 8 \) and \( p = 0 \) or \( p = 1 \), \( x_{g_{p+1}} \) can obviously be selected.

7. For \( p = 2 \), from the results for \( k = 7 \), only \( \bar{G}' \) forms need be enumerated which have no zero column. Table 2.14 lists all possible distinct forms with a suitable \( x_{g_3} \) value shown.

8. For \( p = 3 \), \( \bar{G}' \) has the form

\[
\bar{G}' = \begin{pmatrix}
1 & 0 & 0 & a_{15} & a_{14} & a_{13} & a_{12} & \vdots & a_{11} \\
0 & 1 & 0 & a_{25} & a_{24} & a_{23} & a_{22} & \vdots & a_{21} \\
0 & 0 & 1 & a_{35} & a_{34} & a_{33} & a_{32} & \vdots & a_{31}
\end{pmatrix}
\]  

(2.49)

The enumeration of the \((8, 3)\) cell can be simplified by recognizing similarities with the \((7, 3)\) cell. In (2.49) if the right most column of \( \bar{G}' \) is ignored, the matrix has the same leading diagonal elements and the same dimensions as the \( \bar{G}' \) for the \((7, 3)\) cell. Then for any case where the first seven columns of (2.49) match those of \( \bar{G}' \) for the \((7, 3)\) cell case, the generator, \( x_{g_4} \), used in the \((7, 3)\) cell can also be used for \( x_{g_4}^* \), say, in the \((8, 3)\) cell by letting...
Table 2.13
For \((k, p) = (7, 3)\) All Distinct \(G'\)
Which Satisfy \(\|x_{ai}\| \neq 5 \text{ or } 6\)
and a Satisfactory \(x_{g4}\)

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>(x'_{g1})</th>
<th>(x'_{g2})</th>
<th>(x'_{g3})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3)</td>
<td>(3)</td>
<td>(3)</td>
<td></td>
</tr>
<tr>
<td>(x_{g4})</td>
<td>1 0 0 1 1 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 1 0 1 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 0 1 1 0 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 0 0 1 1 1 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| \(4\)       | \(3\)     | \(3\)     |           |
| \(x_{g4}\)  | 1 0 0 0 1 1 1 |
|             | 0 1 0 0 1 1 0 |
|             | 0 0 1 0 1 0 1 |
|             | 0 0 0 1 0 1 1 |

| \(4\)       | \(4\)     | \(3\)     |           |
| \(x_{g4}\)  | 1 0 0 1 1 1 0 |
|             | 0 1 0 1 1 0 1 |
|             | 0 0 1 1 1 0 0 |
|             | 0 0 0 1 0 1 1 |

| \(4\)       | \(4\)     | \(4\)     |           |
| \(x_{g4}\)  | 1 0 0 1 1 1 0 |
|             | 0 1 0 1 1 0 1 |
|             | 0 0 1 0 1 1 1 |
|             | 0 0 0 1 0 1 1 |
Table 2.14

Remaining Wordlength Combinations for Distinct $G'$ and a Satisfactory $x_{g_3}$ for $(k, p) = (8, 2)$

<table>
<thead>
<tr>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 4</td>
<td>1 0 1 1 1 0 0 0</td>
<td>6 5</td>
<td>1 0 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 0 1 1 1</td>
<td></td>
<td>0 1 0 0 1 1 1 1</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>5 3</td>
<td>1 0 1 1 1 1 0 0</td>
<td>6 6</td>
<td>1 0 1 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 0 0 1 1</td>
<td></td>
<td>0 1 0 1 1 1 1 1</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>5 4</td>
<td>1 0 1 1 1 1 0 0</td>
<td>7 3</td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 0 1 1 1</td>
<td></td>
<td>0 1 1 1 0 0 0 0</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>5 5</td>
<td>1 0 1 1 1 1 0 0</td>
<td>7 4</td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 1 1 1 1</td>
<td></td>
<td>0 1 1 1 1 0 0 0</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>6 3</td>
<td>1 0 1 1 1 1 1 0</td>
<td>7 5</td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 0 0 1 1</td>
<td></td>
<td>0 1 1 1 1 1 0 0</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>6 4</td>
<td>1 0 1 1 1 1 1 0</td>
<td>7 6</td>
<td>1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0 0 1 1 1</td>
<td></td>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
<td>$x_{g_3}$</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
</tbody>
</table>
\[ x^* g_4 = (x g_4, 0) \]

or

\[ x^* g_4 = (x g_4, 1). \]

9. The enumeration of the \((8, 3)\) cell is shown in Table B.1 of Appendix B.

10. The enumeration of the \(k = 9\) case is also in Appendix B.

Corollary 2.5.1. If \(k = 6\) and

1. \(p = 1\) and \(\| x g_1 \| < 5\), or

2. \(p = 2\), \(\| x g_1 \| < 5\), \(\| x g_2 \| < 5\), and

\[ \| x g_1 + x g_2 \| < 5 \]

and a fraction, \(f_i\), is feasible for either case, then a \(2^6-3\) fraction is feasible.

Proof:

1. If case 2 is true, then case 1 must also be true because a satisfactory \(x g_2\) can always be selected. Therefore, only case 2 is considered.

2. Table 2.15 lists all distinct \((x g_1, x g_2)\) pairs which meet the conditions of case 2, and a satisfactory \(x g_3\) is specified for each pair, thereby proving the corollary.
Table 2.15

All Distinct Generator Sets for \((k, p) = (6, 2)\) Such That \(\|x_g^1\| < 5\), \(\|x_g^2\| < 5\), \(\|x_g^1 + x_g^2\| < 6\) Plus a Satisfactory \(x_g^3\)

Wordlengths

<table>
<thead>
<tr>
<th>(x_g^1)</th>
<th>(x_g^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1 1 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>1 1 0 0 1 1</td>
</tr>
<tr>
<td></td>
<td>1 0 1 0 1 0</td>
</tr>
<tr>
<td>x_g^3</td>
<td>1 1 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>1 1 0 0 1 0</td>
</tr>
<tr>
<td></td>
<td>1 0 0 1 0 1</td>
</tr>
<tr>
<td>3</td>
<td>1 1 1 0 0 0</td>
</tr>
<tr>
<td></td>
<td>0 0 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>1 0 0 0 1 1</td>
</tr>
</tbody>
</table>

Corollary 2.5.2. If \(k = 5\), \(p = 1\), \(\|x_g^1\| < 5\), and a fraction \(f_1\) is feasible, then a 25-2 fraction is feasible.

Proof:

1. Two possibilities exist: either

   \(\|x_g^1\| = 4\),

   or

   \(\|x_g^1\| = 3\).

2. If \(x_g^1 = 1 1 1 1 0\), then \(x_g^2 = 1 1 0 0 1\).

3. If \(x_g^1 = 1 1 1 0 0\), then \(x_g^2 = 0 0 1 1 1\), thus
Corollary 2.5.3. Given $A(10, p) = 1$; and for at least one $j$, $a_{ij} = 0$, for $i = 1, \ldots, p$. Then $A(10, p') = 1$ for $p' = p+1, \ldots, 6$.

Proof:

1. a. It is given that $A(10, p) = 1$ and that $a_{ij} = 0$, for $i = 1, \ldots, p$. At no loss in generality, assume that $j = 10$ such that $a_{ij} = 0$, for $i = 1, \ldots, p$.

b. Let $f_j$ be a feasible fraction consisting of the order $2^{10-p}$ set of 10-tuples, $\{x_i\}$. Also let $G$ be the generator set for $f_j$ such that by 1.a,

$$G = [0; G^*]$$

where $G$ is the $p \times 10$ matrix whose rows are the $x_{gi} \in G$. Also let

$$x_i = \begin{pmatrix} a_{i10} \\ x_i^* \end{pmatrix}$$

be considered as a $10 \times 1$ column vector. Since $A(10, p) = 1$, then for some $p \times 1$ binary column vector $C$,

$$G \cdot x_i = C = [0; G^*] \cdot \begin{pmatrix} a_{i10} \\ x_i^* \end{pmatrix} = G^* \cdot x_i^*$$

for all $x_i \in f_j$. This is a sufficient condition for $A(9, p) = 1$. 
2. By theorem 2.5, for \( k = 9 \), additional generators, \( x^*_i \)
where \( i = p + 1, \ldots, 5 \), can always be selected such
that
\[
A(9, p') = 1, \quad p' = p + 1, \ldots, 5.
\]
Therefore generators of the form
\[
xg_i = (0, x^*_i), \quad i = p + 1, \ldots, 5
\]
can always be selected such that \( A(10, p') = 1 \), where
\( p' = p + 1, \ldots, 5 \).

3. a. To demonstrate that a sixth generator can always be
selected under the given conditions such that
\( A(10, 6) = 1 \), consider, at no loss in generality,
the following row echelon form matrix (using simpli­
fied subscript notation)

\[
\begin{pmatrix}
x_{g_1} \\
x_{g_2} \\
x_{g_3} \\
x_{g_4} \\
x_{g_5} \\
x_{g_6}
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & a_{14} & a_{13} & a_{12} & a_{11} \\
0 & 1 & 0 & 0 & 0 & a_{24} & a_{23} & a_{22} & a_{21} \\
0 & 0 & 1 & 0 & 0 & a_{34} & a_{33} & a_{32} & a_{31} \\
0 & 0 & 0 & 1 & 0 & a_{44} & a_{43} & a_{42} & a_{41} \\
0 & 0 & 0 & 0 & 1 & a_{54} & a_{53} & a_{52} & a_{51} \\
0 & 0 & 0 & 0 & 0 & a_{64} & a_{63} & a_{62} & a_{61}
\end{pmatrix}.
\]

Designate \( xg_1 \) as the generator yet to be specified.
As indicated by the above matrix,
\[
xg_1 = (1 0 0 0 0 0 a_{14} a_{13} a_{12} a_{11})
\]
b. Therefore \( x_{g_2}, ..., x_{g_6} \) are the five generators of which \( p \) were given initially with the remaining 5-p values obtained as a result of step 2.

c. To specify \( x_{g_1} \) it remains to specify \( a_{14}, a_{13}, a_{12}, a_{11} \). Consider that there are eleven possible unique permutations for a binary 4-tuple containing at least two nonzero elements (i.e., the two nonzero elements are for the \( R \geq \text{III} \) requirements). Therefore select values for

\[ a_{14}, a_{13}, a_{12}, a_{11} \]

corresponding to one of the six permutations which is not represented by the five 4-tuples

\[ (a_{i4}, a_{i3}, a_{i2}, a_{i1}), \ i = 2, 3, 4, 5, 6. \]

This results in a generator set which satisfies the resolution requirements, thus completing the proof.
CHAPTER III
DESCRIPTION OF ALGORITHM

3.1 INTRODUCTION

The purpose of this chapter is to describe the basic organization of the search algorithm. While doing this, the use of the properties and results of Chapter II is demonstrated.

3.1.1 Feasibility Matrix

The feasibility matrix, Figure 3.1, provides a concise means of conceptualizing the problem and of explaining the organization of the search algorithm. Each cell of the feasibility matrix represents \( k \) ranked factors (vertical axis) and \( 2^{k-p} \) observations (horizontal axis) such that the entire matrix represents all possible combinations of ranked factors and observations under consideration.

The feasibility matrix is partitioned into three regions. The full factorial diagonal consists of those cells having \( k \) factors and \( 2^k \) observations. This diagonal represents the region of division between the replicated full and the fractional factorial regions. In the former region (below the diagonal) there are \( k \) factors and \( m \cdot 2^k \) observations where \( m \) is the number of replications, and in the latter region there are \( k \) factors and \( 2^{k-p} \) observations.
Figure 3.1—Feasibility Matrix with Full Factorial Diagonal, Replicated Full Factorial Region, Fractional Factorial Region, and Nonconstructible Region Designated
such that only \((1/2^p)\) replicates of full factorials are possible. The nonconstructible region represents the situations for which orthogonal fractional factorials in \(k\) factors and \(2^{k-p}\) observations are not constructible because the required inequality condition (2.24), \(k + 1 < 2^{k-p}\), is not satisfied.

To simplify notation, a cell in the fractional factorial region or on the diagonal of the feasibility matrix is referred to as the \((k, p)^{th}\) cell, that is the \(1/2^p\) replicate for \(k\) factors cell.

Given a data base and a ranking of factors, the task of the search algorithm is to determine which cells of the feasibility matrix are feasible. A cell is feasible if at least one fractional factorial in \(k\) factors and \(2^{k-p}\) observations is possible. The \(2^{k-p}\) factorial is feasible if there exists, in the factorial data base, at least one experimental unit for each of the required attribute level or treatment level combinations. At no loss in generality more than one experimental unit per treatment combination can be required.

3.1.2 Path of Search Through Feasibility Matrix

By considering the general sequence in which the cells of the feasibility matrix are investigated by the search algorithm, an introductory overview of the algorithm's organization is provided. A more detailed discussion
follows in the remainder of the chapter. Referring to Figure 3.2 the search begins at the (1, 0) cell and successively considers (2, 0), (3, 0), ..., etc. until it locates the \((k_{\text{min}}, 0)\) cell (\(k_{\text{min}}\) is defined in Section 2.9.2 as the first ranked factor for which a full factorial is not feasible).

Next the search moves horizontally from right to left in the \((k_{\text{min}})^{th}\) row beginning with the \((k_{\text{min}}, 1)\) cell and continuing for larger values of \(p\) [i.e., \((k_{\text{min}}, 2), (k_{\text{min}}, 3), \text{etc.}\)]. Generally, when the first feasible cell in the \((k_{\text{min}})^{th}\) row is determined, further searching of that row ceases and the search continues in the next higher row.

The point at which the search begins in the next higher row depends upon which cells are feasible in the "previous" row. If the \((k, 1)\) cell is feasible [i.e., \(A(k, 1) = 1\)], then the search begins at the \((k + 1, 1)\) cell in the next row (e.g., as is the case in Figure 3.2 when \(k\) increases from 7 to 8). For \(k > 7\) and \(A(k, 1) = 0\), the search in the \((k + 1)^{th}\) row begins in the cell in the same column as the right most feasible cell of the \(k^{th}\) row. Therefore, the general direction of the search is always upward and to the left. For example, in Figure 3.2, cell \((8, 3)\) is the right most feasible cell of the eighth row. Therefore, in the ninth row the search begins at the \((9, 4)\) cell.

The search terminates when conditions indicate the feasibility of all cells has been determined. A variety of
Figure 3.2--The General Sequence in Which Cells are Considered by the Search Algorithm
conditions can indicate this state as discussed in Section 3.5. In Figure 3.2 the search terminates because the maximum number of factors handled by the algorithm, \( k = 10 \), have been considered.

Lastly, the numerous tasks which are performed by the algorithm can be grouped into three categories based upon when the tasks are performed during the search.

First, each time the number of factors \( k \) is incremented from \( k \) to \( k + 1 \) (i.e., going up to the next row in the feasibility matrix), the algorithm:

1. sorts the historical data base, \( Q = \{q_i\} \), in increasing order on the level of \((k + 1)\)th factor,
2. counts the number of experimental units corresponding to each of the \( 2^{k+1} \) factor level combinations from which the empty set, \( E \), and the non-empty set, \( \bar{E} \), are determined, and
3. determines from \( \bar{E} \) the minimum number of times any factor occurs at a zero or nonzero level.

Secondly, for each increment of \( p \) to \( p + 1 \) (i.e., moving from right to left to the next cell in a row of the feasibility matrix) two tasks are performed:

1. the initial values for the set of generators, \( G = \{x_{g_i}\} \), are specified, and
2. the terminating value for each \( x_{g_{i}'} \) is established.

Thirdly, within a \((k, p)\) cell when a generator \( x_{g_{i}} \) is incremented from \( x_{g_{i}} \) to \( (x_{g_{i}} + 1) \) (i.e., incrementing the
binary value of \( x_{g_i} \) several checks are performed on the generator set, \( G \), to determine if \( G \) satisfies the requirements of a generator set. If it does, then the final check is to determine if \( G \) generates a feasible fraction.

The remaining sections of Chapter 3 discuss in more detail the structure of the algorithm demonstrating how the material of Chapter 2 is employed.

3.2 BEGINNING THE SEARCH

3.2.1 Determining \( k_{\min} \)

Figure 3.3 is a flow diagram for determining \( k_{\min} \). As discussed in Section 3.1.2, the algorithm begins at the \((1, \ 0)\) cell of the feasibility matrix sorting the historical database, \( Q \), on the highest priority factor. The first sort counts the number of \( q_i = (a_{i1k_{\max}}, \ldots, a_{i2}, a_{i1}) \) for which \( a_{i1} = 0 \), being \( n_0 \), and the number for which \( a_{i1} = 1 \), being \( n_1 \). The algorithm continues to sort \( Q \) one factor at a time in their ranked order until at least one \( n_i = 0 \). Table 3.1 summarizes the results for the first three sorts.

If at least one \( n_i = 0 \), for any \( i \), then a \( 2^k \) full factorial is not feasible. The quantity \( k_{\min} \) has been defined as the minimum number of ranked factors for which a full factorial is not feasible. Therefore, when the very first \( n_i = 0 \) is encountered, \( k_{\min} \) is determined. Figure 3.4 shows the distribution of feasible cells [i.e., \( A(i, j) = 1 \)]
Sort $Q = \{ q_i \}$ in ascending order on the value of the $k^{th}$ factor preserving the order of equal $q_i$

Any $n_1 = 0$? Yes

$k = k_{\text{max}}$? No $k = k + 1$

$\min k = k$

STOP

Figure 3.3. --Determining $k_{\text{min}}$
Table 3.1
Resulting Information after First Three Sorts

<table>
<thead>
<tr>
<th>Sort</th>
<th>$a_{i1}$</th>
<th>$q_i$</th>
<th>$n_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$a_{i1}$</td>
<td>0</td>
<td>$n_0$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_1$</td>
</tr>
<tr>
<td>2</td>
<td>$a_{i2}$</td>
<td>0</td>
<td>$n_0$</td>
</tr>
<tr>
<td></td>
<td>$a_{i1}$</td>
<td>0</td>
<td>$n_1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_3$</td>
</tr>
<tr>
<td>3</td>
<td>$a_{i3}$</td>
<td>0</td>
<td>$n_0$</td>
</tr>
<tr>
<td></td>
<td>$a_{i2}$</td>
<td>0</td>
<td>$n_1$</td>
</tr>
<tr>
<td></td>
<td>$a_{i1}$</td>
<td>0</td>
<td>$n_2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>$n_3$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_4$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_5$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_6$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>$n_7$</td>
</tr>
</tbody>
</table>
Figure 3.4 -- Known Cell Feasibilities, $A(i,j) = 1$ and Infeasibilities, $A(i,j) = 0$, when $k_{min} = 7$
once $k_{\text{min}}$ is determined ($k_{\text{min}} = 7$ in the example).

### 3.2.2 Determining Empty Cell Set and Minimum Column Count

For all ranked factors $k \geq k_{\text{min}}$ each time $k$ is incremented, the empty cell set

$$E = \{x_{y_i} \mid n_{y_i} = 0\}$$

is constructed. The number of $k$-tuples in $E$ (i.e., the order of $E$) at least doubles every time $k$ is incremented based upon the results of Section 2.8. The set $E$ is used in determining feasibility for the cells in the $k^{th}$ row of the feasibility matrix given a specific generator set, $G$, as discussed in Section 3.3.2.

As the algorithm constructs $E$, it also constructs the nonempty set where

$$\bar{E} = \{x_{z_i} \mid n_{z_i} > 0\}, \quad x_{z_i} = (a_{z_i1}, \ldots, a_{z_2}, a_{z_1}).$$

For each factor $j$ ($j = 1, \ldots, k$), let the number of times (for $\bar{E}$) that $a_{z_i j} = 0$ be denoted by $c_{0j}$ and the number of times $a_{z_i j} = 1$ be denoted by $c_{1j}$. Then define

$$c_{\text{min}} = \min (c_{0j}, c_{1j}), \text{ for all } j.$$  

Also define

$$c_{\text{min}} = \min (c_{\text{min}}_1, c_{\text{min}}_2, \ldots, c_{\text{min}}_k).$$

Therefore, $c_{\text{min}}$ is the minimum number of times any factor appears at a zero or nonzero level in $\bar{E}$. From equation (2.43) each factor in a $2^{k-p}$ fraction occurs $2^{k-p-1}$ times at each level. Therefore, it must be that
The algorithm determines the minimum value for \( p \), denoted \( p_{\text{min}} \), such that (3.1) is satisfied. Then for any \( p' \), for which \( 0 < p' < p_{\text{min}} \), it is true that \( A(k, p') = 0 \).

3.3 SEARCHING A CELL IN THE FRACTIONAL FACTORIAL REGION

Once the value of \( k_{\text{min}} \) is known, the algorithm begins to search the \((k_{\text{min}})^{\text{th}}\) row of the feasibility matrix at the cell determined from inequality (3.1). For \( k > k_{\text{min}} \), other criteria often determine at which cell in the row the search begins.

Two general tasks are associated with searching a cell. First, initial values for the generator set \( G \) must be determined; and terminating values for each \( x_{g_i} \) established.

Secondly, the actual iterating of candidate generator sets, \( \{G\} \), and subsequent testing for feasibility must be performed.

3.3.1 Preparation for the Cell Search

Results of Section 2.7 are used to determine an initial generator set, \( G \), from which the iteration of generators can begin. In many instances candidate generator sets can be eliminated from explicit consideration because the initial generator values can be set at higher (in the binary number sense) than the minimum values. The four possible cases to be considered are:

\[
2^{k-p-1} \leq c_{\text{min}}. \tag{3.1}
\]
1. \( p = 1, k = k_{\text{min}} \)

2. \( p = 1, k > k_{\text{min}} \)

3. \( p \geq 2 \), no previously known \( G \) for the \((k-1, p-1)\) cell, but \( A(k-1, p-1) = 1 \), and

4. \( p \geq 2 \), a \( G \) is known such that 
   \[ [A(k-1, p-1) \mid G] = 1 \].

For the first case (i.e., \( p = 1, k = k_{\text{min}} \)) no prior information is available such that the iteration of \( x_{g_1} \) begins at the smallest possible value (again in the binary number sense), that is

\[ x_{g_1} = (k - 3) 0 \| 111 . \quad (3.2) \]

For the second case (\( p = 1, k > k_{\text{min}} \)) corollary 2.4.2 is used. Let \( x_{g_{\text{min}1}} \) be the minimum (binary value) generator for which it is known that \( A(k - 1, 1) = 1 \). Then

\[ x_{g_1} = (0, x_{g_{\text{min}1}}) . \quad (3.3) \]

For the third case (\( p \geq 2 \), no previous \( G \)) the initial \( x_{g_1} \) must be constructed. Again

\[ x_{g_1} = (k - 3) 0 \| 111 , \quad (3.4) \]

and the remaining generators, \( x_{g_2}, \ldots, x_{g_p} \) are the lowest binary values which collectively satisfy the following three conditions:

1. \( x_{g_p} < x_{g_{p-1}} < \ldots < x_{g_2} < x_{g_1} \) \quad (3.5)

2. \( 3 \leq \| x_{g_i} \| \leq k - p \), for all \( i \), \quad (3.6)

3. \( \| x_{g_i} + x_{g_{i+1}} \| \geq 3 \), for \( i = 1, \ldots, p - 1 \). \quad (3.7)
For the fourth case \((p > 2, G)\) corollary 2.3.1 is applied. Let

\[
G_{\text{min}} = (x_{g_{\text{min}1}}, x_{g_{\text{min}2}}, \ldots, x_{g_{\text{min}p-1}})
\]  

be the minimum binary valued generator set for which it is known that

\[
[A(k-1, p-1) \mid G_{\text{min}}] = 1.
\]

Then the initial generator set for the \((k, p)\)th cell is

\[
x_{g_1} = [0, (x_{g_{\text{min}1}} + 1)] \quad \text{(i.e., the value of } x_{g_{\text{min}1}} \text{ is incremented by 1)}
\]

\[
x_{g_2} = (0, x_{g_{\text{min}1}})
\]

\[
x_{g_3} = (0, x_{g_{\text{min}2}})
\]

\[
\vdots
\]

\[
x_{g_p} = (0, x_{g_{\text{min}p-1}}).
\]

The final task before the search of the cell can begin is to set the terminating values for each \(x_{g_i}\). If \(p = 1\), then from corollary 2.4.1 (page 76) the terminating value for \(x_{g_1}\), denoted \(x_{g_{\text{max}1}}\), is

\[
x_{g_{\text{max}1}} = (k - k_{\text{min}}) \ 0 \ || \ (k_{\text{min}}) \ 1.
\]  

For \(p > 2\), then the terminating values are calculated from equation 2.48 which is:

\[
x_{g_{\text{max}i}} = (i-1) \ 0 \ || \ [k - (i-1) - w(h)] \ 1 \ || \ H(h)
\]

\[
i=1, \ldots, p-1
\]

where \(h = p-i\), and \(w(h)\) and \(H(h)\) are given in Table 2.9.
3.3.2 Generator Iteration - The Cell Search

The total enumeration of the set of all possible generator sets, \( \{G\} \), for a given \( k,p \) is the final step in determining if at least one feasible fraction, \( f_i \), exists. The enumeration is based upon the analogy with the binary number system developed in Section 2.7. Starting from the initial values for each \( x_{g_i} \), as discussed in Section 3.3.1, the \( x_{g_i} \) are systematically incremented. With each increment a series of checks are applied to the \( G \). If all checks are satisfactory, then the partitioning of the empty set, \( E \), by \( G \) is examined to determine feasibility.

Two cases of generator iteration are considered: the simpler case of \( p = 1 \), and the more difficult case of \( p \geq 2 \). Figure 3.5 illustrates the \( p = 1 \) case. Only two conditions must be satisfied by \( x_{g_1} \) to be a generator as indicated in Figure 3.5. The iteration stops and the search of the \( (k, 1)^{th} \) cell is completed when either \( A(k, 1) = 1 \) or when every possible \( x_{g_1} \leq x_{g_{\text{max}_1}} \) has been considered.

For \( p \geq 2 \), exhaustive enumeration of \( \{G\} \) (the set of all possible generator sets) is a difficult task as is discussed in Section 2.1. Two problems exist. First, use of the binary number analogy of Section 2.7 results in many candidate generator sets which are not independent or do not
Figure 3.5. -- Iteration of $x_{g_1}$ for the $p = 1$ Case
meet the minimum resolution requirement (i.e., $R \geq \text{III}$). Secondly, enumeration of $\{G\}$ is redundant as discussed in Section 2.4.3 because: 1) it is only necessary to consider once each alias subgroup, $A$; but 2) each $A$ is considered several times with enumeration of $\{G\}$. Since the computational burden required to check feasibility for a given $G$ can be large, particularly when the empty set $E$ is large and $p$ is large, it is desirable to consider each $A$ only once.

As discussed in Section 3.3.1, use of $x_{\text{gmax}}$ as terminating values reduces redundant enumeration. Also use of the $x_{\text{gmin}}$ to construct the initial $G$ eliminates consideration of some $G$ for which it is known without checking that $[A(k, p) \mid G] = 0$. All remaining methods to improve the efficiency of the enumeration are divided into two phases. The first phase is a series of checks on $x_{g}$ and on subsets of $G$ as $x_{g}$ is incremented. The second phase is a series of checks upon $G$ before feasibility is tested.

Figure 3.6 is a flow diagram of the generator iteration based upon the binary number analogy. This generator iteration exhaustively enumerates $\{G\}$. The iteration always maintains inequality (2.31),

$$x_{g_p} < x_{g_{p-1}} < \ldots < x_{g_2} < x_{g_1}.$$  

It also conducts the first phase of checks as are now discussed in the explanation of the iteration logic. As an aid to understanding the following material, the reader may wish
to first refer to Appendix F which contains a simple example of the generator iteration sequence.

Beginning at node (L1) in Figure 3.6 (page 112), assume the current \( G \) has already been tested and determined to be not feasible such that the next value of \( G \) is needed. Each \( x_{g_i} \in G \) is compared with its maximum value. If \( x_{g_i} = x_{g_{\max i}} \) for all \( i \), then the enumeration is complete and \( A(k, p) = 0 \). If not, then for the first \( x_{g_i} \) (in ascending order of \( i \)) for which

\[
x_{g_i} < x_{g_{\max i}}
\]

(the usual case), that \( x_{g_i} \) is incremented using binary arithmetic at point (L2).

The iteration logic considers one generator at a time—the \( i^{th} \) one where \( i = 1, \ldots, p \). The logic is such that control is transferred to node (L3) (page 113) each time \( x_{g_i} > x_{g_{\max i}} \) or each time a generator other than \( x_{g_1} \) is incremented.

At node (L4) (page 113) is \( i = p \), then \( x_{g_p} > x_{g_{\max p}} \). This condition indicates that \( \{G\} \) has been exhaustively enumerated because all possible sets

\[
(x_{g_{p-1}}, x_{g_{p-2}}, \ldots, x_{g_1})
\]

have been considered for each possible value of \( x_{g_p} \). Therefore, \( A(k, p) = 0 \). If, at node (L4) it is true that \( i < p \), then consideration switches to
Figure 3.6 -- Flow Diagram of Generator Iteration for $p \geq 2$ Case
If:
1. \(3 \leq |x_{g_1}| \leq k-p\)
2. \(3 \leq |x_{g_1} + x_{g_2}|\)
3. \(#\text{factors} \geq k_3\)

Yes \(x_{g_1} = x_{g_1} + 1\)
No \(x_{g_1} = x_{g_1} + 1\)

Is \(x_{g_1} > x_{g_{\text{max}_1}}\)?
Figure 3.6 -- continued
Figure 3.6 -- continued
which is then incremented [branch (L5)].

When the test at node (L3) is false, a second test at node (L6) determines if control transfers to branch (L7) for $i = 1$ or branch (L8) (page 114) if $i > 1$. In branch (L8) an acceptable value for $x_{g_i}$ is determined as long as

$$x_{g_i} < x_{g_{\text{max}_i}}.$$

At node (L9) the wordlength of $x_{g_i}$ is checked. At node (L10) if $i < p-1$, then $||x_{g_i} + x_{g_{i+1}}||$ is checked. If in either case the wordlength is unsatisfactory, the $x_{g_i}$ is incremented to a new value for which the series of checks begins again. However if both wordlengths satisfy the tests, then the initial checks on $x_{g_i}$ are complete, and control passes to node (L11) where a new value for $x_{g_i-1}$ and $i$ is decreased by one (i.e., the former $x_{g_i-1}$ now becomes $x_{g_i}$).

If at node (L10) $i \geq p-1$, a check for $i = p-1$ follows. If this check is false, then $i = p$; and control transfers to (L11) where $i$ is decreased as previously discussed. Therefore $i = p-1$ when neither $i < p-1$ or $i = p$; and control transfers to branch (L12) (page 115) which determines, based upon $x_{g_{p-1}}$ and $x_{g_p}$, whether the alias subgroup represented by the current value of $G$ has been previously considered in the enumeration of $\{G\}$. The first check in
branch (L12) determines if the wordlength of \((x_{g_{p-1}} + x_{g_p})\) is satisfactory. If not, then \(x_{g_{p-1}}\) is incremented [i.e., control transfers back to branch (L8) through connecting point (F)]. Next at node (L13) the binary value of \((x_{g_{p-1}} + x_{g_p})\) is compared with \(x_{g_p}\). If \((x_{g_{p-1}} + x_{g_p}) < x_{g_p}\) and \(|| x_{g_{p-1}} + x_{g_p} || \leq k - p\), then generator sets with \(x^*_{g_p} = x_{g_{p-1}} + x_{g_p}\), say, have been previously considered in the iteration. Therefore no new alias subgroups can occur for the \(x_{g_{p-1}}, x_{g_p}\) pair, so \(x_{g_{p-1}}\) is incremented. Otherwise the value of \(x_{g_{p-1}}\) is retained, and the value of \(i\) is decreased by one at node (L11).

When, for all \(i > 1\), the conditions of branch (L8) are satisfied and when \(x_{g_i} < x_{g_{\text{max}_i}}\), then, \(i = 1\) and control is transferred to branch (L7) of Figure 3.6 (page 111). This branch conducts the final checks which complete the first phase of checks on \(G\).

In summary, the first phase of checks are:

1. \(x_{g_i} \leq x_{g_{\text{max}_i}}\) \hspace{1cm} \(i = 1, \ldots, p\)
2. \(3 \leq || x_{g_i} || \leq k - p\) \hspace{1cm} \(i = 1, \ldots, p\)
3. \(3 \leq || x_{g_i} + x_{g_{i+1}} ||\) \hspace{1cm} \(i = 1, \ldots, p - 1\)
4. number of factors at nonzero level \(\geq k_3\).

The fourth check is based on results which are summarized in Table 2.6. When all four sets of conditions are satisfied,
control is transferred to the row echelon form (REF) reduction which begins the second phase of checks on G.

Figure 3.7 shows the sequence of the second phase of checks. First G is reduced to G' using Bailey's (1959) methods as discussed in Section 2.5. If G does consist of p independent $x_{g_i}$, then the rank of G is p. Secondly, a test for minimum resolution of three is conducted ($R > \text{III}$) again using Bailey's technique.

If the above two requirements are met, then G constitutes an independent set of generators with at least minimum resolution. The third requirement is that the current G' does not represent a previously considered alias subgroup. Every alias subgroup, A, has a unique G' as discussed in Section 2.9.1. Using a hashing function approach (Knuth, 1973), G' is compared to the contents, say G*, of a particular array location. The array location is based upon the value of a function whose argument is calculated from G'. Simplifying the situation somewhat, if

$$G' = G^*,$$

then the alias subgroup, A, generated by G' has been previously considered.

If G' represents a new A, then the final check is to determine feasibility. For each $x_{y_i} \in E$, the empty set, a binary p-tuple, $(c_1, c_2, \ldots, c_p)$, is calculated as follows:
START: Test the Candidate Generator Set, G.

Go To Generator Iteration, \( i = 1 \)

**REF**
Reduce \( G \) to Row Echelon Form, \( G' \).
Is Rank = \( p \)?

Yes

Is \( R \geq \text{III} \)?

Yes

New Alias Subgroup?

No, Previously Considered

Yes

\( A(k, p) = 1 \)?

No, Infeasible

STOP

No, Rank < \( p \)

No, \( R < \text{III} \)

No, Previously Considered

Yes

Figure 3.7. -- Final Series of Checks on \( G \) to Determine Feasibility
The calculation of \( p \)-tuples stops when:

1. all \( 2^p \) possible \( p \)-tuples have occurred (this indicates that each fractional factorial, \( f_i \in \mathcal{F} \), the family of fractions, has at least one empty cell, \( x_{y_i} \)) or

2. all \( x_{y_i} \) have been considered in which case
   \[ A(k, p) = 1. \]

Lastly, from Figure 3.7, whenever any check fails, then \( G \) is infeasible and the enumeration of generator sets begins again. The enumeration continues until completion unless \( A(k, p) = 1 \) at some point. Otherwise, \( A(k, p) = 0 \). Once \( A(k, p) \) is determined, the search of the \( (k, p) \)th cell is completed.

### 3.4 INTER-CELL DEPENDENCIES

The search algorithm does not explicitly consider all of the cells in the fractional factorial region. Rather, once the state of a cell is determined, this result often can be used to specify the feasibility of other cells. Table 3.2 summarizes the relationships.

Relationship 1 (the diagonal rule) of Table 3.2 applies to half replicates. Given \( A(k, 1) = 0 \) then all half
Table 3.2

Inter-cell Dependencies

Given $A(k, p) = 0$, then

1. if $p = 1 \Rightarrow A(k + 1, 1) = 0$ (Diagonal Rule)
2. $A(k + 1, p + 1) = 0$ (Column Rule)

Given $A(k, p) = 1$, then

3. $A(k, p') = 1$ for all $p' > p$
   such that $k + 1 < 2^{k-p'}$ (Row Rule)

   Conditions
   a. $8 \leq k \leq 9$
   b. $k = 7$:
      i. $p < 3$ and $p' = p+1$, ..., 3 or
      ii. $p \leq 3$ and $\|x_{a_i}\| \neq 5$ or 6
   c. $k = 10$: if for at least one $i$
      $a_{g_{ij}} = 0$ for $j=1$, ..., $p$
   d. $5 \leq k \leq 6$ if $\|x_{a_i}\| \leq 4$
replicates for larger values of \( k \) are not feasible. The diagonal rule is the result of Theorem 2.4.

Relationship 2 (the column rule) of Table 3.2 is based upon Theorem 2.3. Once \( A(k, p) = 0 \), then the column above the \((k, p)\)th cell consists of all zeros.

Relationship 3, the row rule, has several cases. The general result is that once \( A(k, p) = 1 \), then all cells to the left in the row are feasible. No special conditions are needed for \( 8 \leq k \leq 9 \) (Theorem 2.5). For \( k = 7 \), two possibilities exist. If for \( p < 3 \) \( A(7, p) = 1 \), then \( A(7, p') = 1 \) for \( p' = p + 1, \ldots, 3 \). In addition if the wordlength condition is satisfied for the \((7, p)\) cell (i.e., \( \| x_{a_i} \| \neq 5 \) or 6), then \( A(7, 4) = 1 \) also (Theorem 2.5).

For the row rule to apply for \( k = 10 \), at least one factor must be at the zero level for each \( x_{g_i} \in G \) (i.e., for some \( i \), \( a_{g_i,j} = 0 \) for all \( j \) by Corollary 2.5.3). The chances of this condition occurring are aided by the fact that the generators are enumerated by beginning at the smallest possible values, thereby increasing the possibility of 0's in \( G \). If \( 5 \leq k \leq 6 \), then the row rule holds (based upon Corollaries 2.5.1 and 2.5.2) only if for each member of the alias subgroup, \( x_{a_i} \), that

\[
\| x_{a_i} \| \leq 4
\]

Figure 3.8 depicts the implications of each of the three rules.
Figure 3.8--Implications of the Column, Diagonal and Row Rules
3.5 TERMINATING THE SEARCH

Based upon the column rule, the search terminates if an entire row in the fractional factorial region of the feasibility matrix is not feasible.

The search also terminates if the only feasible cell in a row is the smallest possible replicate such that

\[ A(k, p') = 1 \] but for \( k + 1 \) and \( p' + 1 \)

\[ k + 1 > 2^{(k+1)-(p'+1)} \]

This occurs for \((k, p) = (7, 4)\) and \((3, 1)\).

Based upon equation (3.1) if for the largest possible value of \( p' \)

\[ 2^{k-p'-1} > c_{\text{min}} \]

then even the smallest fractional replicate cannot be feasible. Therefore, the entire row is zero and the search stops.

The search stops if the factors to consider in the historical data base have been exhausted or if the tenth factor, the maximum considered by the algorithm, is completed.

Lastly, the search stops if the allotted computational time on the computer is exceeded (this issue is discussed in Chapter 4).
Chapter IV consists of three sections. Section 4.1 discusses the attributes of the algorithm design which improve the efficiency of the search. In Section 4.2 the search results for independently generated random data sets are considered to assess general operating characteristics of the algorithm. In the final section, Section 4.3, an application of the search algorithm to a highway intersection data base is demonstrated.

Attributes of the algorithm which improve the efficiency of the search can be divided into two categories being:

1. use of relationships which allow implicit enumeration for some cells of the feasibility matrix, and

2. use of relationships which permit generator sets to be skipped in the iteration within a cell.

The first category consists of four relationships. The row rule, with additional qualifications as discussed in Section 3.4, states that given \( A(k, p) = 1 \), then

\[
A(k, p') = 1, \text{ for } p' > p,
\]

where an upper bound for \( p' \) is determined by the constraint
Therefore once a cell is determined to be feasible, then further searching in that row of the feasibility matrix is not required. To take full advantage of the row rule the search always begins at that possibly feasible $k^{th}$ row cell for which $p$ is the smallest value. If this cell is found to be feasible, then the maximum number of other cells in the $k^{th}$ row have been implicitly determined feasible also. In addition the right most cells of the feasibility matrix have fewer generator sets to enumerate because generator sets have $p$ generators and $p$ decreases going from left to right in a feasibility matrix row.

The column rule, which is:

$$A(k + 1, p + 1) = 0 \quad \text{given} \quad A(k, p) = 0,$$

results in a column of zeros above the first zero cell as discussed in Section 3.4. Therefore the smaller the $k$ for which $A(k, p) = 0$, the greater the number of cells which are eliminated from consideration. If an entire row is not feasible, then the search stops.

The diagonal rule, similar to the column rule, is:

$$A(k + 1, 1) = 0 \quad \text{given} \quad A(k, 1) = 0.$$

The diagonal rule together with the column rule results in every cell which is above and to the right of an infeasible half replicate cell, also not being feasible.

The last rule in the first category is

$$2^{k-p-1} \leq c_{\min}$$
where \( c_{\text{min}} \) is the minimum factor level frequency count as defined in Section 3.2.2. The above rule eliminates consideration of the right most cells in the \( k^{\text{th}} \) row for which \( p \) is too small (i.e., making the number of required observations too large) such that the above inequality is not satisfied.

The second category of relationships consists mainly of corollaries of the column and diagonal rules. Suppose in the iteration for the \((k, p)^{\text{th}}\) cell, the first feasible generator set is

\[
G = (x_{g_1}, x_{g_2}, \ldots, x_{g_p})
\]

such that \( A(k, p) = 1 \). Then in consideration of the \((k + 1, p + 1)^{\text{th}}\) cell the initial generator set, by Corollary 2.3.1, is \( G^* \), say, where

\[
G^* = (x^*_{g_1}, x^*_{g_2}, \ldots, x^*_{g_{p+1}})
\]

\[
= [(0, x^*_{g_1} + 1), (0, x^*_{g_1}), (0, x^*_{g_2}), \ldots, (0, x^*_{g_p})].
\]

Two important points are at issue. First, for any generator set, say \( G_i \), such that \( G_i < G^* \) (here using the binary number analogy), then \( G_i \) is known to be not feasible and therefore can be skipped. Secondly, because \([A(k, p) \mid G] = 1\), there is a good possibility that \( A(k + 1, p + 1) = 1 \) for a generator set for which

\[
x^*_{g_i} = (0, x^*_{g_i}) \quad i=2, \ldots, p+1.
\]

Therefore only a few iterations of the first generator,
x^* _g1_, may be necessary before a feasible generator set results.

4.2 SEARCH OF RANDOM DATA SETS

Random data sets having different total number of observations have been generated by the process discussed in Appendix C. The basic characteristic of each data set is that it represents a uniformly distributed random sample of the set of all possible binary 10-tuples (i.e., the 1024 unique binary 10-tuples). The search results for these data sets are summarized in Table 4.1.

As indicated in Table 4.1, some data sets have been searched for different values of k_{max}, which is the absolute maximum number of factors the algorithm is limited to considering in one search. While this strategy does not result in independent estimates of search times within the repeated data base, it does allow for more efficient use of the available computing resources. This is a significant concern given the costs of the runs for the longer search times.

Several points are of interest with respect to the results of Table 4.1. First when the search terminates for a value of k, say k', such that k' < k_{max}, then for all values of k > k', the cells are implicitly determined to be not feasible. For example consider data set 2 of Table 4.1. The search terminated at the (k, p) = (6, 3) cell with
Table 4.1

Search Times and Terminating Cell Results, Grouped by Data Sets, for Random Uniformly Generated Data Sets of Varying Size

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Size</th>
<th>k_{max}</th>
<th>Terminating Cell Result</th>
<th>Search Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>10</td>
<td>A(4, 1) = 0</td>
<td>6.25 seconds</td>
</tr>
<tr>
<td>2</td>
<td>32</td>
<td>6</td>
<td>A(6, 3) = 0</td>
<td>11.2 seconds</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>6</td>
<td>A(6, 3) = 0</td>
<td>9.37 seconds</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
<td>6</td>
<td>A(6, 3) = 0</td>
<td>12.1 seconds</td>
</tr>
<tr>
<td>5</td>
<td>64</td>
<td>6</td>
<td>A(6, 3) = 1</td>
<td>11.6 seconds</td>
</tr>
<tr>
<td>5</td>
<td>64</td>
<td>10</td>
<td>A(7, 4) = 0</td>
<td>21.2 seconds</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>6</td>
<td>A(6, 3) = 1</td>
<td>14.1 seconds</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
<td>10</td>
<td>A(7, 4) = 0</td>
<td>23.2 seconds</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
<td>10</td>
<td>A(7, 4) = 0</td>
<td>3.82 minutes</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>10</td>
<td>A(7, 4) = 1</td>
<td>4.02 minutes</td>
</tr>
<tr>
<td>9</td>
<td>64</td>
<td>10</td>
<td>A(7, 4) = 0</td>
<td>41.8 seconds</td>
</tr>
<tr>
<td>10</td>
<td>73</td>
<td>10</td>
<td>A(7, 4) = 0</td>
<td>2.23 minutes</td>
</tr>
<tr>
<td>11</td>
<td>128</td>
<td>6</td>
<td>A(6, 2) = 1</td>
<td>13.0 seconds</td>
</tr>
<tr>
<td>11</td>
<td>128</td>
<td>7</td>
<td>A(7, 3) = 1</td>
<td>1.70 minutes</td>
</tr>
<tr>
<td>11</td>
<td>128</td>
<td>8</td>
<td>A(8, 4) = ?*</td>
<td>7 minutes</td>
</tr>
<tr>
<td>12</td>
<td>135</td>
<td>7</td>
<td>A(7, 3) = 1</td>
<td>63.3 seconds</td>
</tr>
<tr>
<td>12</td>
<td>135</td>
<td>8</td>
<td>A(8, 4) = ?</td>
<td>8 minutes</td>
</tr>
<tr>
<td>13</td>
<td>241</td>
<td>6</td>
<td>A(6, 0) = 1</td>
<td>6.02 seconds</td>
</tr>
<tr>
<td>13</td>
<td>241</td>
<td>7</td>
<td>A(7, 2) = 1</td>
<td>21.0 seconds</td>
</tr>
<tr>
<td>13</td>
<td>241</td>
<td>8</td>
<td>A(8, 3) = ?</td>
<td>9 minutes</td>
</tr>
<tr>
<td>14</td>
<td>512</td>
<td>8</td>
<td>A(8, 2) = 1</td>
<td>41.8 seconds</td>
</tr>
<tr>
<td>14</td>
<td>512</td>
<td>9</td>
<td>A(9, 3) = ?</td>
<td>3 minutes</td>
</tr>
<tr>
<td>15</td>
<td>832</td>
<td>9</td>
<td>A(9, 2) = 1</td>
<td>1.52 minutes</td>
</tr>
<tr>
<td>15</td>
<td>832</td>
<td>10</td>
<td>A(10, 3) = ?</td>
<td>7 minutes</td>
</tr>
<tr>
<td>16</td>
<td>1024</td>
<td>10</td>
<td>A(10, 0) = 1</td>
<td>15.0 seconds</td>
</tr>
</tbody>
</table>

* A question mark (?) indicates the terminating cell feasibility was not resolved in the indicated search time.
\(A(6, 3) = 0\). Therefore the entire \(k = 6\) row was not feasible, which also meant that no cell was feasible for \(k > 6\).

For data set 8 the search terminated with a feasible cell, that is \(A(7, 4) = 1\), for \(k < k_{\text{max}}\). The reason for this apparent inconsistency (i.e., why did not the search continue) was that in this particular case \(A(7, p') = 0\) for \(p' < 4\). Therefore for the search to have continued, the next cell to search would have been the \((8, 5)\) cell, which is in the nonconstructible region. Hence the search stopped.

For several of the data sets the search terminated at the \((7, 4)\) cell with \(A(7, 4) = 0\). For data sets 5, 6, and 9 the search times were 21.2, 23.2, and 41.8 seconds, respectively. However, for data sets 7 and 10 the times were 3.82 minutes and 2.23 minutes, respectively. The reason for the large differences in times between the two groups of data sets was a function of the "path" the search took to reach the \((7, 4)\) cell for each group. For the first group (i.e., the lower times) the search proceeded from the \((6, 3)\) cell to the \((7, 4)\) cell. However, for the second group the path was from the \((7, 3)\) cell to the \((7, 4)\) cell. Therefore the longer search times of the second group reflected the fact that more cells had to searched to reach the \((7, 4)\) cell than were searched for the first group of data sets.

Further interpretation of the results in Table 4.1 are considered in greater detail in Chapter V. The general
trends are:

1. Six or fewer factors can be searched at very low cost (i.e., low search time) essentially independent of the data base size.

2. As data base size increases above 64, the search times go up significantly with seven factors being the maximum number one can be fairly certain of searching in reasonable amounts of computing time (less than five to ten minutes, say).

3. For large data bases (e.g., greater than 500) the search times can be small for the first one or two cells which must be searched, but the algorithm becomes bogged down when attempting to totally enumerate an infeasible cell for \( k \geq 8 \) and \( p \geq 3 \).

4.3 HIGHWAY INTERSECTION EXAMPLE

As an example of the application of the search algorithm to a practical problem, consider again the highway intersection example first discussed in Chapter 1. The question under investigation concerns in what manner driver deceleration patterns on approaches to intersections are systematically influenced, if any, by attributes of the intersections? It is desired to investigate the question by observing the behavior of actual road users in the natural setting of public highway intersections.

A data base describing the 431 signalized rural intersections in the State of Ohio is available to aid in the selection of intersection sites for observations. The intersection attributes to be considered, their ranking, and their coding to 0, 1 levels is shown in Table 4.2.
### Table 4.2
Ranking and Coding of Intersection Attributes

<table>
<thead>
<tr>
<th>Rank</th>
<th>Factor</th>
<th>Level 0</th>
<th>Level 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>approach speed limit (mph)</td>
<td>&lt; 50</td>
<td>&gt; 50</td>
</tr>
<tr>
<td>2</td>
<td>traffic flow</td>
<td>lower 33%</td>
<td>upper 33%</td>
</tr>
<tr>
<td>3</td>
<td>total accidents over previous five years</td>
<td>lower 33%</td>
<td>upper 33%</td>
</tr>
<tr>
<td>4</td>
<td>wet accidents (5 years)</td>
<td>&lt; .25</td>
<td>&gt; .25</td>
</tr>
<tr>
<td>5</td>
<td>horizontal curvature</td>
<td>not present</td>
<td>present</td>
</tr>
<tr>
<td>6</td>
<td>grade</td>
<td>not present</td>
<td>present</td>
</tr>
<tr>
<td>7</td>
<td>lanes</td>
<td>2</td>
<td>&gt; 2</td>
</tr>
</tbody>
</table>

Note in Table 4.2 that the coding of factors 2 and 3 results in sampling from the extremes on each factor. The intent of the coding is to increase the likelihood that any strength of association either factor has with the observed measures of driver deceleration behavior shall not go undetected. The cost of employing the partitioning of the factors towards their extremes is that the data base is reduced from 431 to 190 intersections.

The search of the intersection data base for 190 sites terminated in 19.4 seconds at A(6, 3) = 1. The set of feasible generators is
\[ x_{g_1} = 100110 \]
\[ x_{g_2} = 010011 \]
\[ x_{g_3} = 001101 \]  

Figure 4.1 summarizes which cells are feasible given the 190 intersections in the data base. From Figure 4.1, no fraction is feasible for all seven factors while a \( 2^{6-3} \) fraction is feasible for the first six factors and a \( 2^{5-2} \) for the first five. A full factorial is possible if only the first four factors are considered.

Table 4.3 lists the eight \( 6 \)-tuples for the feasible \( 2^{6-3} \) fraction and their frequency of occurrence in the data base. It is interesting to point out that for the family of eight fractions specified by the generator set of equations (4.1), only that fraction shown in Table 4.3 is feasible. Also for five of the eight \( 6 \)-tuples only one intersection exists having the necessary attributes. This can pose a serious problem because for many reasons a particular intersection may not be usable. For example there may be no satisfactory location to position an unobtrusive observer, or the geometry of a particular site may not be compatible with the instrumentation being used. In addition, even if the intersection is observable, there is always a significant probability that a data collection session will not be successful. Therefore the researcher must weigh these issues when selecting a particular fraction for study.
Figure 4.1 -- Feasibility Matrix for Highway Intersection Data Base with 190 Observations
Table 4.3

Listing of 8 Factor Level Combinations, $x_i$, Which Comprise a $2^6-3$ Fraction and the Number of Times Each $x_i$ Occurs in the Intersection Data Base (190 Observations)

<table>
<thead>
<tr>
<th>$x_i = (a_{i6}, a_{i5}, a_{i4}, a_{i3}, a_{i2}, a_{i1})$</th>
<th>$n_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 0 1 1</td>
<td>3</td>
</tr>
<tr>
<td>0 0 1 1 0 0</td>
<td>2</td>
</tr>
<tr>
<td>0 1 0 0 1 0</td>
<td>1</td>
</tr>
<tr>
<td>0 1 0 1 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 0 0 0 0 0</td>
<td>2</td>
</tr>
<tr>
<td>1 0 0 1 1 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 1 0 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 1 1 1 0</td>
<td>1</td>
</tr>
</tbody>
</table>

One alternative to the above problem is to change the coding used in the original data base, Table 4.2, such that all 431 intersections are included by losing the extreme partitionings for factors 2 and 3. For this case the search required 3.62 minutes computation time. The results for this larger data set are shown in Figure 4.2.

Interpreting Figure 4.2, seven factors in eight observations or six factors in either sixteen or eight observations are now feasible. Table 4.4 contains the results for a $2^7-4$ fraction while Table 4.5 contains the results for a $2^6-2$ fraction. For the $2^7-4$ fraction, four attribute combinations still have only one potential site. However for
Figure 4.2—Feasibility Matrix for the Highway Intersection Data Base with 431 Observations
the $2^{6-2}$ case, only four of the sixteen are limited to one corresponding intersection.

Table 4.4

Listing of 8 Factor Level Combinations, $x_i$, Which Comprise a $2^{7-4}$ Fraction with the Number of Times Each $x_i$ Occurs in the Intersection Data Base (431 Observations)

<table>
<thead>
<tr>
<th>$x_i = (a_i7, a_i6, a_i5, a_i4, a_i3, a_i2, a_i1)$</th>
<th>$n_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 1 0 0 0</td>
<td>11</td>
</tr>
<tr>
<td>0 0 1 0 1 1 0</td>
<td>3</td>
</tr>
<tr>
<td>0 1 0 0 1 0 1</td>
<td>1</td>
</tr>
<tr>
<td>0 1 1 1 0 1 1</td>
<td>1</td>
</tr>
<tr>
<td>1 0 0 1 1 1 1</td>
<td>16</td>
</tr>
<tr>
<td>1 0 1 0 0 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 0 0 0 1 0</td>
<td>3</td>
</tr>
<tr>
<td>1 1 1 1 1 0 0</td>
<td>1</td>
</tr>
</tbody>
</table>
Table 4.5

Listing of 16 Factor Level Combinations, $x_i$, Which Comprise a $2^{6-2}$ Fraction with the Number of Times Each $x_i$ Occurs in the Intersection Data Base (431 Observations)

$$x_i = (a_{i6}, a_{i5}, a_{i4}, a_{i3}, a_{i2}, a_{i1})$$

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>$n_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 1 0</td>
<td>27</td>
</tr>
<tr>
<td>0 0 0 1 0 1</td>
<td>20</td>
</tr>
<tr>
<td>0 0 1 0 1 1</td>
<td>8</td>
</tr>
<tr>
<td>0 0 1 1 0 0</td>
<td>9</td>
</tr>
<tr>
<td>0 1 0 0 0 0</td>
<td>5</td>
</tr>
<tr>
<td>0 1 0 1 1 1</td>
<td>2</td>
</tr>
<tr>
<td>0 1 1 0 0 1</td>
<td>2</td>
</tr>
<tr>
<td>0 1 1 1 1 0</td>
<td>5</td>
</tr>
<tr>
<td>1 0 0 0 1 0</td>
<td>4</td>
</tr>
<tr>
<td>1 0 0 1 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 0 1 0 1 1</td>
<td>2</td>
</tr>
<tr>
<td>1 0 1 1 0 0</td>
<td>2</td>
</tr>
<tr>
<td>1 1 0 0 0 0</td>
<td>2</td>
</tr>
<tr>
<td>1 1 0 1 1 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 1 0 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 1 1 0 0</td>
<td>1</td>
</tr>
</tbody>
</table>
CHAPTER V
GENERAL CONSIDERATIONS IN THE USE OF THE ALGORITHM

5.1 SOLUTION APPROACH WITHOUT ALGORITHM

In the spring of 1976 the author was involved in the problem of selecting intersections as has been discussed in the highway example of Chapters I and IV. At the time ad hoc procedures were used in an attempt to find feasible $2^{6-2}$ fractions, but the effort was soon abandoned when no promising results appeared at all likely. Given no other alternatives the approach finally used was to determine, by routine sorting procedures, the largest feasible full factorial, which was four factors. Then using the standard techniques for constructing fractions, half replicates for $x_{g_1} = 1 1 1 1$

were examined.

Due to the absence of a formal definition of the search problem in the literature and because typical problems can require several minutes of high speed digital computer computation time, it is believed that if other researchers have tried to solve the problem heuristically, they probably were not successful. The search algorithm presented in this work provides a very practical alternative which often results in inexpensive solutions for typical problems.
However, there are also circumstances when the algorithm cannot complete the search as to be discussed. Lastly, it is important to note that while the algorithm represents a successful logical formulation of the problem, it is not an optimal formulation [see Knuth (1973) for discussion of optimality measures for algorithms]. Therefore it is quite likely that more efficient algorithms could be developed.

5.2 UPPER BOUNDS BY CELL FOR TOTAL ENUMERATION

As k and p get large, the most time consuming task for the algorithm is to explicitly determine a cell to be infeasible [i.e., \( A(k, p) = 0 \)] by totally enumerating every possible alias subgroup, \{A\}. This is the key issue in limiting the progress of the algorithm in mapping the feasibility matrix.

To establish rough orders of magnitude of the maximum number of enumerations possibly required for each cell of the feasibility matrix, calculations as discussed in Section 2.4.5 have been performed and are summarized in Figure 5.1. The figure clearly demonstrates the severity of the problem with increasing k and p.

As a quantification of the implications of Figure 5.1, several random uniformly distributed data bases*, \( \{Q_i\} \), have been searched to obtain representative estimates of the magnitude of computer processing time required as a function of

*See Appendix C for description of data base generation.
Figure 5.1--Upper Bounds for Maximum Possible Number of Unique Alias Subgroups by Cell
$k_{\text{max}}$ and the size of the data base, denoted as $N_i$.

The approach is to determine for each $Q_i$ the processing time, $t_i$, for increasing values of $k_{\text{max}}$ until one of the following occurs:

1. the search terminates because no potentially feasible cells remain to search given $Q_i$, or
2. the $t_i$ exceeds the maximum time as specified by the researcher.

This blocking on $Q_i$ for different values of $k_{\text{max}}$ does not result in independent estimates of processing time within a data base. However, the computation costs of the investigation are a substantial amount such that the savings warrant the blocking approach.

Figure 5.2 is a plot of time, $t$, versus data base size, $N$, for different values of $k_{\text{max}}$. These empirical results agree with the implications of Figure 5.1, that is $t_i$ increases as $k$ and $p$ increase.

Figure 5.2 suggests the following generalizations. For small $Q_i$ (i.e., $N \leq 64$) the maximum number of feasible factors is limited by $N$ (assuming for $Q_i$ that $k_{\text{max}} \geq 8$). In other words for $N \leq 64$ a search of $Q_i$ is very likely to be completed in a reasonable time resulting in the maximum number of feasible factors being determined at some value less than $k_{\text{max}}$. The relationship between $N$ and the maximum number of likely feasible factors, based upon the Figure 5.2 results, is listed in Table 5.1.
Figure 5.2—Plot of Times to Complete Search Versus Data Base Size with Terminating Cell Value Indicated
Table 5.1

Maximum Number of Factors Likely To Be Feasible Given N

<table>
<thead>
<tr>
<th>N</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>3</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
</tr>
<tr>
<td>64</td>
<td>7</td>
</tr>
</tbody>
</table>

Conversely for large \(Q_i\) (i.e., \(N \geq 128\)) the search time becomes very large when determining a cell to be not feasible for \(k > 8\) and \(p > 3\). For these cases the time limit is often exceeded before the maximum number of feasible factors can be determined. One strategy in this situation is to limit \(k_{\text{max}}\) to 6 or 7, say, such that feasibility is determined very quickly because 6 or 7 factors are usually feasible when \(N \geq 128\).

Figure 5.3, summarizing the data of Figure 5.2, depicts contours of approximate search times for different values of \(k_{\text{max}}\). For example with \(k_{\text{max}} = 6\) the search time probably would not exceed 10 to 15 seconds for any value of \(N\). No contours are shown for \(k_{\text{max}} > 7\) and \(N \leq 64\) because all cells were implicitly determined to be infeasible in this region. That is the searches always terminated because the feasibility of all cells was determined and none were feasible for \(k > 7\).

The data bases used for the study represent uniform random selections from the 1024 possible 10-tuples. Also by
Figure 5.3--Contours of Times to Complete Search
the manner in which the data bases were constructed (see Appendix C) no 10-tuple could be repeated. Therefore each factor on the average occurred equally often at the zero and nonzero levels, and none of the factors covaried. In general a typical historical data base would have repeated values and some, if not substantial, covariation among factors. Therefore the historical data base would probably require a larger number of observation than would the uniform random data base for the same level of feasible fractions.

5.3 STRATEGIES TO PERFORM INITIAL SEARCH

To perform the initial search of a data base, \( Q_i \), the selection of a processing time limit, \( t_{\text{max}} \), can be based upon \( k_{\text{max}} \) and/or \( N_i \), or a sequential search strategy can be employed. Based upon the results of Figure 5.3 if \( k_{\text{max}} \leq 6 \), then a \( t_{\text{max}} \) of one minute, say, would probably be more than adequate. If \( k_{\text{max}} \geq 8 \) and \( N_i \) is either very large (i.e., \( N_i > 1000 \)) or very small (i.e., \( N_i < 64 \)), then again a one minute time limit would probably be sufficient. If the combination of \( k_{\text{max}} \geq 8 \) and \( N_i \) fall somewhere in the middle of the range, say 500, then it may prove useful to perform the search in two stages. First search for \( k_{\text{max}} = 8 \) and \( t_{\text{max}} = 3 \) minutes. Then if no cell for \( k = 7 \) is feasible, the search will very probably be completed and no further searching is necessary. If, however, one or more cells are
feasible for $k = 7$, then the search would proceed to the eighth row and probably exceed the time limit while in that row if any $A(8, p) = 0$ for $p \geq 3$. At this point the researcher would have to decide, based upon the results of the first search, whether to employ a second search to attempt to finish the enumeration of the cell which was under consideration when the first search exceeded $t_{\text{max}}$. To aid his decision he can compare the final value of $x_{g_p}$ to $x_{g_{\text{max}p}}$ from the first search because once $x_{g_p} = x_{g_{\text{max}p}}$, then $A(k, p) = 0$. If $x_{g_p}$ only has a few more suitable values (i.e., $3 \leq ||x_{g_p}|| \leq k-p$), before $x_{g_p} = x_{g_{\text{max}p}}$, then the algorithm has a good chance, particularly for $k = 8$, to complete the search. However, if $x_{g_p}$ is near its minimum value, then the search would probably not be completed if a total enumeration were required [i.e., $A(k, p) = 0$]. Figure 5.4 illustrates these considerations.

5.4 INITIAL PLANNING ISSUES

It is almost certain that at least minor if not substantial manipulation of the original historical data will be required to transform that data to meet the requirements of the algorithm. Strategies concerning different aspects of this transformation are now discussed.
\[ k_{\text{max}} \leq 8 \]

\[ N_i < 64 \]

\[ t_{\text{max}} = 1 \text{ minute} \]

\[ N_i > 1000 \]

\[ t_{\text{max}} = 1 \text{ minute} \]

\[ t_{\text{max}} = 3 \text{ minutes, first search, then:} \]

1. If \( A(7,p) = 0 \) for all \( p \), then no need for second search.

2. If \( A(8,p) \) was not completely enumerated then:
   a. If \( x_g = x_{g_{\text{max}}p} \), then may get a successful determination of \( A(8,p) \) in a reasonable amount of time.
   b. If \( x_g \neq x_{g_{\text{max}}p} \), then a second search could very likely be inconclusive for moderate \( t_{\text{max}} \) (i.e., 7-10 minutes)

**Figure 5.4--Decision Tree for Initial Search**
5.4.1 Partitioning the Data

The basic issue is that a continuous variable, or a discrete variable with several levels, must be transformed to a variable having only two levels -- 0, 1. For example in the highway intersection problem all sites with approach speed limits less than or equal to 50 miles per hour were designated low approach speed sites while those with greater than 50 mile per hour speed limits were designated high approach speed sites. For each variable or factor, considered individually, the requirements of the transformation are:

1. the transformation must not submerge, in the simplification to two levels, any influence the factor may have on the dependent measure, and

2. the transformation should, as nearly as possible, divide the historical data base in half on the two levels of the factor.

The second requirement reflects the property that each full and fractional factorial has each factor appear equally often at its zero and nonzero levels. Therefore to increase the probabilities of finding feasible fractions, the equal splits are usually preferred.

To aid in making the variable transformations, scatter diagrams and frequency counts are often very helpful. Statistical programming software packages such as SAS76 (Barr, et. al., 1976) allow the researcher to quickly obtain such information. In addition SAS76 is also well suited for
performing the actual data manipulations to produce the transformed data.

Another consideration in the transformation of the data is the covariation of the factors. If two factors, say $f_1$ and $f_2$, highly covary (e.g., they have a sample correlation coefficient of 0.95, say), then for nearly all members of the historical data base the factors occur in either $(0, 0)$ or $(1, 1)$ pairs because of the strong covariation. Therefore it could be difficult to find feasible fractions containing both $f_1$ and $f_2$ as factors. To minimize this problem one factor, say $f_1$, could be ranked high with the other ranked very low. Alternatively a redefinition of factors could combine $f_1$ and $f_2$ into one factor, $f'_1$, where

$$f'_1 = \frac{1}{2}(f_1 + f_2)$$

as an example. In general, an examination of the partial correlations of all factors may aid in the ranking of factors or suggest combining two or more factors into one composite factor.

5.4.2 Partitioning on Extremes

Often the purpose of an experiment is to detect the presence, if any, of covariation between the designated independent and dependent variables. The probability of successfully detecting such covariation, when it exists, can be increased by partitioning one or more historical data base factors on their extreme values rather than using a
collectively exhaustive partitioning. Such a partitioning is employed in the highway intersection example of Section 4.2 where for both the second and third ranked factors the zero-level is the lower 33% of the factor values and the one-level is the upper 33% of the factor values. Therefore the intersections corresponding to the middle 33% of the factor values for either or both factors are excluded from the historical data base.

The following example demonstrates why the partitioning on extremes increases the probability of detection. For this example let $x$ be a continuous independent variable and $y$ the dependent variable. Then consider the first order regression model

$$y = a + bx + e$$

where $e$ is a normally distributed random variable with an expectation of zero and a variance of $\sigma^2 = 1$. The slope parameter, $b$, is a measure of the covariation between $x$ and $y$. To estimate $b$, suppose eight independent observations of $y$ are taken for $x$ equal to $x_{\text{low}}$ and eight observations taken for $x$ equal to $x_{\text{high}}$, where $x_{\text{high}} > x_{\text{low}}$. Then the width of a 90% confidence interval for $b$ (i.e., the interval is a measure of the uncertainty in the estimate of $b$ and therefore is also an estimate of the uncertainty in the degree of covariation between $x$ and $y$) is:

$$\text{interval width} = \frac{2}{(x_{\text{high}} - x_{\text{low}})}$$

Figure 5.5, a plot of the interval width as a function of
Figure 5.5 -- 90% Confidence Interval Width for the Slope Parameter, \( b \), as a Function of \( x_{\text{high}} - x_{\text{low}} \)
(x_{\text{high}} - x_{\text{low}}), clearly demonstrates the advantage to be gained by partitioning a factor on its extreme values (i.e., larger values of x_{\text{high}} - x_{\text{low}}).

An additional benefit of partitioning on extremes is the reduced probability of misclassifying a site as a zero-level instead of a one-level or vice versa. For example with the highway intersection study the partitioning on approach velocity speed limit is set at a division point of 50 miles per hour. Therefore an intersection designated as a high velocity site may, upon observation, turn out to actually be a low velocity site. However, if the partitioning of sites were on extremes, say less than 45 miles per hour or greater than 55 miles per hour, then the chances of misclassification should be less.

Of course a limiting factor to the degree of extreme partitioning which can be employed, is the subsequent reduction in historical data base size. In the intersection case, partitioning on the extreme thirds of two factors reduced the number of intersections from 431 to 190.

5.4.3 Missing Data

It is not uncommon in many applications to have an historical data base \( Q = \{q_j\} \) for which individual members, \( q_j \), have one or more missing attribute values. For \( q_j \) to be a member of a feasible fraction or full factorial, all levels for the \( k \) ranked factors under consideration must be
known for \( q_i \). Therefore if, say, half of the attribute values are not known for the \((k + 1)\)th ranked factor, then the effective size of \( Q \) is reduced by a factor of two when increasing the number of ranked factors under consideration from \( k \) to \( k + 1 \).

The above problem suggests a factor ranking strategy based upon missing data properties alone. The first step would be to determine the frequency of missing data for each factor and then select an initial ranking based upon relative factor importance and the missing data characteristics. For this ranking a search is conducted. If the results are unsatisfactory (i.e., the maximum number of factors which can be considered is judged too low), then the researcher can:

1. rerank the factors and try again,
2. change one or more data transformations to obtain a better balance between 0, 1 levels, or
3. attempt to fill in the missing holes in the data.

This latter alternative becomes more appealing if the higher ranked factors (those thought a priori to be most important) have severe missing data problems.

What is considered a satisfactory search result by the researcher depends upon the specific circumstances of the investigation. If, for example, the maximum number of observations that time and/or funds will allow is sixteen,
than a full factorial in six factors, requiring 64 observations, is not useful. However if a $2^6$ full factorial is feasible, then it can be used to obtain feasible $2^{6-1}$, $2^{6-2}$, or $2^{6-3}$ fractions. Also if there is very little reason to rank any of several factors high or low (e.g., an exploratory study), then the researcher might wish to include more factors in the study hoping to get some evidence on as many factors as possible. At this stage, main effects and perhaps two way interactions are of most interest to the researcher. Therefore he would probably prefer, for example, a $2^{9-3}$ fraction over a $2^6$ full factorial. Also the $2^6$ factorial has 44 degrees of freedom associated with three way and higher interaction effect estimates.

Lastly, it is interesting to note that if $A(k_{\text{max}}, p) = 0$, where $k_{\text{max}}$ is the maximum number of factors that can be considered, then no reordering of the ranking of the same $k_{\text{max}}$ factors would change the infeasibility condition.

5.4.4 Large Number of Sites per Factor Level Combination

If the number, $n_i$, is large for every factor level combination, $x_i$, of a feasible fraction, $f_i$, then the researcher has several possible options at the expense of the large values for $n_i$. As demonstrated in the highway example of Chapter IV, he can partition one or more factors on their extremes thereby increasing the opportunity to detect any
patterns of covariation.

A second alternative is to reduce the size of the historical data base, Q, by controlling nuisance factors which equivalently reduces the $n_i$. Again, for the highway example, the original data base of over 16,000 intersections was reduced to one of 431 signalized intersections.

5.4.5 Small Number of Sites

Repeating the discussion of Chapter IV, if one or more $n_i$ are small, the researcher may have serious doubts about being able to actually obtain a successful observation for every $x_i$ in the fraction $f_i$. Therefore he may elect to consider less than the maximum feasible $k$ in return for larger values of $n_i$. Also, especially for $k \leq 7$, the minimum acceptable $n$, $n_{\min}$, could be set at a higher value, such that the search would only be for fractions with all $n_i \geq n_{\min}$.

5.4.6 Blocking a Fraction

Lastly, once a feasible fraction, $f_i$, is obtained, then the fraction can be run in $r$ blocks of $2^{k-p}/r$ observations each where $r$ equals a power of 2. Since the set of generators, $G$, is known for $f_i$ from the output of the algorithm, then the blocking terms can be selected in the usual manner as when the researcher has direct control over the factor levels [see, for example, John (1971) or Box and Hunter (1961)].
CHAPTER VI

SUMMARY AND RECOMMENDATIONS

The problem of determining a feasible fractional factorial, when the experimenter cannot directly control some or all of the factor levels, is believed, for lack of evidence to the contrary in the literature, to never have been attacked by other than ad hoc procedures. This dissertation presents a systematic design strategy incorporating a computerized algorithm to efficiently solve the above problem for many practical cases.

6.1 SUMMARY

Chapter I introduces, as motivation of the dissertation research, the problem of selecting experimental units when attributes of those units are considered as independent variables. In addition to this issue the proposed design strategy is also applicable to situations when the researcher does have control over the factor levels, but due to constraints imposed by the system under study, some factor level combinations are not obtainable. For example in the study of a manufacturing plant some combinations of production levels, worker levels, and product mix may not be physically possible to attain. Therefore the researcher wants to know if a feasible fraction exists for the combinations of factor
levels which he can achieve.

Chapter I also describes the three basic components of the design strategy: a historical data base, a factorial data base, and the search algorithm; and generally how they collectively comprise the proposed design strategy.

In Chapter II the pertinent literature on orthogonal fractional factorials is reviewed. In addition several theorems and corollaries are developed which have direct bearing upon the efficiency of the algorithm.

An overview of the structure of the search algorithm, somewhat detailed in parts, is provided in Chapter III. This includes the rationale for the sequence in which cells of the feasibility matrix are searched and also the inter-cell dependencies regarding feasibility and infeasibility.

Chapter IV summarizes attributes of the algorithm which relate to its efficiency, and the performance of the algorithm is considered for the case of random, uniformly generated data bases. The latter portion of Chapter IV is devoted to an application of the design strategy to the highway intersection example.

Lastly in Chapter V general considerations in the use of the algorithm are discussed. The results for the random, uniformly distributed data bases are summarized in a manner which suggests how a researcher could effectively use the algorithm while not incurring excessive computational costs. The chapter concludes with a discussion of several issues
concerning the use of the algorithm and the initial planning of the experiment.

6.2 RECOMMENDATIONS

The key recommendation concerning the algorithm developed in this work addresses the problem of the excessive computational burden required to determine a cell to be infeasible for \( k \geq 8 \) and \( p \geq 3 \). Direct enumeration of row echelon form generator sets, \( \{G'\} \), could provide some improvement over enumeration of all generator sets, \( \{G\} \). However this would probably have only minor impact upon the computation times. Rather, a radically different strategy from total enumeration must be used if substantial improvement is to be obtained. A possible alternative strategy could be using partial enumeration results together with examination of the specific empty cell structure under consideration to make probabilistic estimates of the chances for feasibility if the search is continued. Then the search could be continued or discontinued based upon the outcome of an appropriate decision rule.

If the requirement for orthogonal fractions with each factor at two levels is relaxed, then many different possibilities exist depending upon what type of fractional replicate is specified. See, for example, Addelman (1972), Plackett and Burman (1946), or John (1971) for discussion of various types of fractional replicates. Probably the most
useful extension would be to the case of nonorthogonal replicates with each factor at two levels because fractions consisting of observations in integer multiples of four are possible [Plackett and Burman (1946)].

6.3 CONCLUSIONS

A very practical tool has been developed to aid in both the design and execution of experiments having factorial treatment combinations where some or all of the factor level combinations are not under direct experimenter control. As research efforts become more complex, then often more factors need to be simultaneously considered and direct control over factor levels in the traditional sense is not possible (e.g., research conducted in natural settings). For these cases more design aids, as the one developed in this work, are needed to improve the quality of such efforts.
APPENDIX A

Definition of Row Echelon Form
Let \( \mathbf{G} \) be a \( p \times k \) matrix composed of \( p \) binary \( k \)-tuple row vectors, \( x_i \), such that

1. 
\[
\begin{pmatrix}
    x_1 \\
    x_2 \\
    \vdots \\
    x_p
\end{pmatrix}
\]

and,

2. \( x_i = (a_{ik}, a_{ik-1}, \ldots, a_{i2}, a_{i1}) \)

where \( a_{ij} = 0, 1 \) (note: \( j \) subscript decreases from left to right).

Then \( \mathbf{G} \) is in row echelon form if:

1. no \( x_i \) is a null vector,

2. \( a_{iq_i} \) is the first (i.e., the left most) nonzero element in \( x_i \) then
\[
q_1 > q_2 > \ldots > q_p, \text{ and}
\]

3. \( a_{iq_j} = 0 \) for all \( i \neq j \).

Define \( a_{iq_i} \) as the leading element of \( x_i \).
APPENDIX B
Enumeration of Cases for Selection of (p+1)th Generator
The completion of the enumeration for the (8,3) cell is presented in Table B.1. The remaining tables are for the (9,2), (9,3) and (9,4) cases respectively. Again, no set of generators are enumerated for $k=9$ if: 1) any column of $\overline{G}'$ is all zeros or 2) if a suitable solution has been demonstrated in the $k=8$ enumerations. Also only distinct generator sets are considered. If, as in equation 2.39, $\overline{G}'$ is partitioned such that

$$\overline{G}' = [I : H],$$

then two generator sets are considered distinct if the $H$ matrix of one generator set cannot be obtained from the $H$ matrix of the other by a sequence of column and/or row operations. Also for given wordlengths, combinations of generators are not listed which obviously can have the same $(p+1)^{th}$ generator as a case already considered.
Table B.1

Enumeration of Remaining Distinct Generator Combinations for the (8,3) Cell with Selection of a Suitable Fourth Generator, $x_{g_4}$

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 5 5</td>
<td>10011111</td>
<td>01011110</td>
<td>00111101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10011000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 5 4</td>
<td>10011111</td>
<td>01011100</td>
<td>00111001</td>
<td>00111100</td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td>10000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 4 3</td>
<td>10011111</td>
<td>01011100</td>
<td>00110001</td>
<td>00100011</td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td>10001100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10011111</td>
<td>01011001</td>
<td>00110001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 3 3</td>
<td>10011111</td>
<td>01011000</td>
<td>00110001</td>
<td>00100101</td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td>10000110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 5 5</td>
<td>handled by $k=7$ case</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 5 4</td>
<td>10011110</td>
<td>01011101</td>
<td>00111100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wordlengths</td>
<td>( x_{g1} )</td>
<td>( x_{g2} )</td>
<td>( x_{g3} )</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>5 5 3</td>
<td>10011110</td>
<td>10011110</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011101</td>
<td>01011101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00110001</td>
<td>00100011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000011</td>
<td>10000101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 4 4</td>
<td>10011101</td>
<td>10011101</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011100</td>
<td>01011001</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00110001</td>
<td>00111000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10010010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 4 3</td>
<td>10011110</td>
<td>10011110</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011100</td>
<td>01011001</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00110001</td>
<td>00111000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000011</td>
<td>10001001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 3 3</td>
<td>10011110</td>
<td>10011110</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011000</td>
<td>01011000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00110001</td>
<td>00111000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 4 4</td>
<td>handled by ( k=7 ) case</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 4 3</td>
<td>10011100</td>
<td>10011100</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011010</td>
<td>01011010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00110001</td>
<td>00100101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000011</td>
<td>10000101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 4 3</td>
<td>10011100</td>
<td>10011100</td>
<td>( x_{g4} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011010</td>
<td>01011010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00100011</td>
<td>00100011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wordlengths</td>
<td>$x_{g_1}$</td>
<td>$x_{g_2}$</td>
<td>$x_{g_3}$</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>4 3 3</td>
<td>10011100</td>
<td>10011100</td>
<td>10011100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01011000</td>
<td>01010010</td>
<td>01010010</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00100011</td>
<td>00110001</td>
<td>00110001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td>10000111</td>
<td>10000111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$x_{g_4}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 3 3</td>
<td>10011000</td>
<td>10011000</td>
<td>10011000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01010100</td>
<td>01010100</td>
<td>01010100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00100011</td>
<td>00100011</td>
<td>00100011</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000110</td>
<td></td>
<td>$x_{g_4}$</td>
<td></td>
</tr>
</tbody>
</table>

Table B.1--continued
Table B.2

Enumeration of Remaining Distinct Generator Combinations for the (9,2) Cell with Selection of a Suitable Third Generator, $x_{g_3}$

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 3</td>
<td>101111111</td>
<td>011000001</td>
<td>000111000</td>
<td>$x_{g_3}$</td>
<td></td>
</tr>
<tr>
<td>7 3</td>
<td>101111110</td>
<td>011000001</td>
<td>000111000</td>
<td>$x_{g_3}$</td>
<td></td>
</tr>
<tr>
<td>6 3</td>
<td>101111100</td>
<td>010000011</td>
<td>000111000</td>
<td>$x_{g_3}$</td>
<td></td>
</tr>
<tr>
<td>8 7</td>
<td>101111111</td>
<td>011111110</td>
<td>001110000</td>
<td>$x_{g_3}$</td>
<td></td>
</tr>
</tbody>
</table>
Table B.3

Enumeration of Remaining Distinct Generator Combinations for the (9,3) Cell with Selection of a Suitable Fourth Generator, $x^g_4$

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x^g_1$</th>
<th>$x^g_2$</th>
<th>$x^g_3$</th>
<th>$x^g_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 6</td>
<td>100111111</td>
<td>010111110</td>
<td>001111101</td>
<td>000111000</td>
</tr>
<tr>
<td>7 6 5</td>
<td>100111111</td>
<td>010111110</td>
<td>001111100</td>
<td>001111001</td>
</tr>
<tr>
<td></td>
<td>100110000</td>
<td>100110000</td>
<td></td>
<td>$x^g_4$</td>
</tr>
<tr>
<td>7 6 4</td>
<td>100111111</td>
<td>010111110</td>
<td>001111000</td>
<td>001110001</td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000001110</td>
<td></td>
<td>$x^g_4$</td>
</tr>
<tr>
<td>7 6 3</td>
<td>100111111</td>
<td>010111110</td>
<td>001100000</td>
<td>001100001</td>
</tr>
<tr>
<td></td>
<td>000001110</td>
<td>000011100</td>
<td></td>
<td>$x^g_4$</td>
</tr>
<tr>
<td>6 6 3</td>
<td>100111110</td>
<td>010111101</td>
<td>001100001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>000011100</td>
<td></td>
<td></td>
<td>$x^g_4$</td>
</tr>
<tr>
<td>$\mathbf{a_B}$</td>
<td>$\mathbf{b_B}$</td>
<td>$\mathbf{c_B}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01011000</td>
<td>01001100</td>
<td>$4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000111000</td>
<td>$5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0000111100</td>
<td>$4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0000111100</td>
<td>$3$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0000111110</td>
<td>$4$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table B.3--continued**
<table>
<thead>
<tr>
<th>4.6x</th>
<th>5.6x</th>
<th>6.6x</th>
<th>7.6x</th>
</tr>
</thead>
<tbody>
<tr>
<td>01110000</td>
<td>00011100</td>
<td>00110010</td>
<td>00011110</td>
</tr>
<tr>
<td>10001001</td>
<td>10001110</td>
<td>10001101</td>
<td>10001101</td>
</tr>
<tr>
<td>10000100</td>
<td>10000100</td>
<td>10000100</td>
<td>10000100</td>
</tr>
<tr>
<td>00011101</td>
<td>00011101</td>
<td>00011101</td>
<td>00011101</td>
</tr>
</tbody>
</table>

Table B.3--continued

171
### Table B.3—continued

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 3 3</td>
<td>100111000</td>
<td>010100100</td>
<td>001000011</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>000001110</td>
</tr>
<tr>
<td>3 3 3</td>
<td>100110000</td>
<td>010001100</td>
<td>001000011</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>000011010</td>
</tr>
</tbody>
</table>
Table B.4
Enumeration of Remaining Distinct Generator Combinations for the (9,4) Cell with Selection of a Suitable Fifth Generator, x_{g_5}
(Note: the leading 4x4 identity matrix is not shown)

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>x_{g_1}</th>
<th>x_{g_2}</th>
<th>x_{g_3}</th>
<th>x_{g_4}</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 5 5 5</td>
<td>11111</td>
<td>11110</td>
<td>11101</td>
<td>11011</td>
</tr>
<tr>
<td></td>
<td>000011100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 5 5 4</td>
<td>44422</td>
<td>44332</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td></td>
<td>11101</td>
<td>11101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td>11010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 5 5 3</td>
<td>44322</td>
<td>43332</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td></td>
<td>11101</td>
<td>11101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td>10010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Numerals are counts of number of 1's appearing in each column; serve as aid to keeping track of different cases
<table>
<thead>
<tr>
<th>( x_{g1} )</th>
<th>( x_{g2} )</th>
<th>( x_{g3} )</th>
<th>( x_{g4} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>44321</td>
<td>43331</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>10010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( x_{g5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>43322</td>
</tr>
<tr>
<td>33332</td>
</tr>
<tr>
<td>11111</td>
</tr>
<tr>
<td>11111</td>
</tr>
<tr>
<td>11110</td>
</tr>
<tr>
<td>11110</td>
</tr>
<tr>
<td>11000</td>
</tr>
<tr>
<td>11000</td>
</tr>
<tr>
<td>10001</td>
</tr>
<tr>
<td>00011</td>
</tr>
<tr>
<td>000001111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( x_{g5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>44222</td>
</tr>
<tr>
<td>11111</td>
</tr>
<tr>
<td>11110</td>
</tr>
<tr>
<td>11001</td>
</tr>
<tr>
<td>11000</td>
</tr>
<tr>
<td>000001111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( x_{g5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
</tr>
<tr>
<td>43321</td>
</tr>
<tr>
<td>11111</td>
</tr>
<tr>
<td>11110</td>
</tr>
<tr>
<td>11000</td>
</tr>
<tr>
<td>10100</td>
</tr>
<tr>
<td>000000111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( x_{g5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>33322</td>
</tr>
<tr>
<td>11111</td>
</tr>
<tr>
<td>11110</td>
</tr>
<tr>
<td>11000</td>
</tr>
<tr>
<td>00101</td>
</tr>
<tr>
<td>000001011</td>
</tr>
</tbody>
</table>

| \( x_{g5} \) |
Table B.4--continued

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>(x_{g_1})</th>
<th>(x_{g_2})</th>
<th>(x_{g_3})</th>
<th>(x_{g_4})</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 4 4 4</td>
<td>44222</td>
<td>43322</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11010</td>
<td>11010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11001</td>
<td>10101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 4 4 4</td>
<td>33332</td>
<td>33332</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11010</td>
<td>10011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00111</td>
<td>01110</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000001101</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 4 4 3</td>
<td>44221</td>
<td>43321</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11010</td>
<td>11010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000100111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0000010011</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000001011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>43222</td>
<td>33332</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11010</td>
<td>11010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10001</td>
<td>00110</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000001011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>43222</td>
<td>32332</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10011</td>
<td>10011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td>00110</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000010011</td>
<td>000001011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000010011</td>
<td>000010011</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(x_{g_5}\)
Table B.4--continued

<table>
<thead>
<tr>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>43321</td>
<td>33331</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>00110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000001101</td>
<td>000001011</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>33322</td>
<td>32323</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>10001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101</td>
<td>00101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000001011</td>
<td>000001011</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>42221</td>
<td>33311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10010</td>
<td>01100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010001</td>
<td>011000001</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>33221</td>
<td>32222</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01010</td>
<td>00011</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010010</td>
<td>000001110</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010001</td>
<td></td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
</tbody>
</table>
### Table B.4--continued

<table>
<thead>
<tr>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>42221</td>
<td>33311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10010</td>
<td>01100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010001</td>
<td>011000001</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>33221</td>
<td>32222</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111</td>
<td>11111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01010</td>
<td>00011</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010010</td>
<td>000001110</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010001</td>
<td></td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>44322</td>
<td>43332</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11101</td>
<td>11101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11011</td>
<td>11011</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>10110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100010001</td>
<td>100010001</td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
<tr>
<td>33333</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111100000</td>
<td></td>
<td>$x_{g_5}$</td>
<td></td>
</tr>
</tbody>
</table>
Table B.4--continued

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>(x_{g_1})</th>
<th>(x_{g_2})</th>
<th>(x_{g_3})</th>
<th>(x_{g_4})</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 5 5 3</td>
<td>44222</td>
<td>43322</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11101</td>
<td>11101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11011</td>
<td>11011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td>10100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td>(x_{g_5})</td>
<td></td>
</tr>
</tbody>
</table>

5 5 4 4

| 44321       | 43322       |             |             |
| 11110       | 11110       |             |             |
| 11101       | 11101       |             |             |
| 11100       | 11100       |             |             |
| 11010       | 10011       |             |             |
| 100011001   | 100011001   | \(x_{g_5}\) |             |

33332

| 11110       |             |             |             |
| 11101       |             |             |             |
| 11011       |             |             |             |
| 01011       |             |             |             |
| 100011001   |             | \(x_{g_5}\) |             |

5 5 4 3

<p>| 44311       | 43321       |             |             |
| 11110       | 11110       |             |             |
| 11101       | 11101       |             |             |
| 11100       | 11100       |             |             |
| 11000       | 10010       |             |             |
| 000000111   | 000000111   | (x_{g_5}) |             |</p>
<table>
<thead>
<tr>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>33322</td>
<td>33322</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td>11101</td>
<td>11101</td>
<td>11100</td>
<td>00111</td>
</tr>
<tr>
<td>00011</td>
<td>11000</td>
<td>10001001</td>
<td>10001001</td>
</tr>
<tr>
<td>43311</td>
<td>43211</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td>11101</td>
<td>11101</td>
<td>11000</td>
<td>11000</td>
</tr>
<tr>
<td>10100</td>
<td>10010</td>
<td>00000111</td>
<td>00000111</td>
</tr>
<tr>
<td>33321</td>
<td>33222</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td>11101</td>
<td>11101</td>
<td>11000</td>
<td>11000</td>
</tr>
<tr>
<td>00110</td>
<td>00011</td>
<td>00001011</td>
<td>00001111</td>
</tr>
<tr>
<td>44221</td>
<td>43321</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td>11010</td>
<td>11010</td>
</tr>
<tr>
<td>11001</td>
<td>10101</td>
<td>00001001</td>
<td>00001011</td>
</tr>
<tr>
<td>43222</td>
<td>33331</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td>11001</td>
<td>11010</td>
</tr>
<tr>
<td>10011</td>
<td>00111</td>
<td>00001010</td>
<td>10001001</td>
</tr>
</tbody>
</table>

Table B.4--continued
Table B.4--continued

<table>
<thead>
<tr>
<th>$x_{g_1}$</th>
<th>$x_{g_2}$</th>
<th>$x_{g_3}$</th>
<th>$x_{g_4}$</th>
<th>$x_{g_5}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 4 4 4 4</td>
<td>33322</td>
<td>11110</td>
<td>11100</td>
<td>11001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 4 4 3 3</td>
<td>43221</td>
<td>42222</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td>11010</td>
<td>10011</td>
</tr>
<tr>
<td></td>
<td>000010111</td>
<td>100010101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 4 3 3 3</td>
<td>33321</td>
<td>33222</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td>11010</td>
<td>11001</td>
</tr>
<tr>
<td></td>
<td>000010011</td>
<td>000010101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>33222</td>
<td></td>
<td>11110</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00111</td>
<td></td>
<td>11001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11000</td>
<td></td>
<td>11000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>000010101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 4 3 3 3</td>
<td>43211</td>
<td>42221</td>
<td>11110</td>
<td>11110</td>
</tr>
<tr>
<td></td>
<td>11100</td>
<td>11100</td>
<td>11000</td>
<td>10010</td>
</tr>
<tr>
<td></td>
<td>000000111</td>
<td>000000111</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table B.4--continued

<table>
<thead>
<tr>
<th>( x_{g1} )</th>
<th>( x_{g2} )</th>
<th>( x_{g3} )</th>
<th>( x_{g4} )</th>
<th>( x_{g5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>33311</td>
<td>33311</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101</td>
<td>00101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001011</td>
<td>00000111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33221</td>
<td>32321</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00111</td>
<td>10100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001101</td>
<td>00001101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32222</td>
<td>32222</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>10111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>10001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011</td>
<td>11000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001101</td>
<td>00001101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42221</td>
<td>32221</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td>11110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>10100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>00011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000000111</td>
<td>000001101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22222</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11110</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000010111</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_{g_1}$</td>
<td>$x_{g_2}$</td>
<td>$x_{g_3}$</td>
<td>$x_{g_4}$</td>
<td>$x_{g_5}$</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>4 4 4 4</td>
<td>43221</td>
<td>42222</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 4 4 4</td>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11010</td>
<td>11010</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11001</td>
<td>10101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10110</td>
<td>10011</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00000111</td>
<td>00000111</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 4 4 4 3  | 33321    | 33222    |
|          | 11100    | 11100    |
|          | 11010    | 11010    |
|          | 10101    | 10101    |
|          | 01110    | 01011    |
|          | 00000111 | 00000111 |

| 4 4 4 3  | 44111    | 43211    |
|          | 11100    | 11100    |
|          | 11010    | 11010    |
|          | 11001    | 11001    |
|          | 11000    | 10100    |
|          | 00000111 | 00000111 |

| 4 4 4 3  | 42221    | 33311    |
|          | 11100    | 11100    |
|          | 11010    | 11010    |
|          | 10110    | 10101    |
|          | 10001    | 01100    |
|          | 00000111 | 00000111 |

| 4 4 4 3  | 33221    | 32222    |
|          | 11100    | 11100    |
|          | 11010    | 11010    |
|          | 10101    | 10101    |
|          | 01010    | 00011    |
|          | 00000111 | 00001101 |

<table>
<thead>
<tr>
<th>4 4 4 3</th>
<th>$x_{g_5}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 4 4 3</td>
<td>$x_{g_5}$</td>
</tr>
</tbody>
</table>
### Table B.4--continued

<table>
<thead>
<tr>
<th>Wordlengths</th>
<th>$x_{g1}$</th>
<th>$x_{g2}$</th>
<th>$x_{g3}$</th>
<th>$x_{g4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 4 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43111</td>
<td>42211</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>11100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11010</td>
<td>11010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>10100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>10001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000111</td>
<td>00000111</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 33211       | 32221    |
| 11100       | 11100    |
| 11010       | 11010    |
| 11000       | 10100    |
| 00101       | 00011    |
| 00000111    | 00001101 |

| 32221       | 22222    |
| 11100       | 11100    |
| 01011       | 11010    |
| 10100       | 00101    |
| 10010       | 00011    |
| 00001101    | 00001101 |

| 22222       |          |
| 11100       |          |
| 01011       |          |
| 00101       |          |
| 10010       |          |
| 00001110    |          |

| 4 3 3 3     |          |          |          |          |
| 32211       | 22221    |
| 11100       | 11100    |
| 11000       | 11000    |
| 10100       | 00110    |
| 00011       | 00011    |
| 00001110    | 00001101 |

| $x_{g5}$    |          |          |          |
| 00001111    |          |

| 00000101    |          |

| 00000110    |          |

| 00001101    |          |

| 00001101    |          |

| 00001110    |          |

| 00001110    |          |

| $x_{g5}$    |          |          |          |
Table B.4—continued

<table>
<thead>
<tr>
<th>$x_{g1}$</th>
<th>$x_{g2}$</th>
<th>$x_{g3}$</th>
<th>$x_{g4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>12222</td>
<td>11100</td>
<td>01010</td>
<td>00101</td>
</tr>
<tr>
<td>00011</td>
<td>000010111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>41111</td>
<td>32111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>11000</td>
<td>10100</td>
<td>10100</td>
</tr>
<tr>
<td>10010</td>
<td>10010</td>
<td>10001</td>
<td>01001</td>
</tr>
<tr>
<td>00000111</td>
<td>00000111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22211</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000010011</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX C

Description of Generation of Random Data Sets
Figure C.1 demonstrates the method used to generate random, uniformly distributed historical data bases of varying size. If approximately \( p \) percent of the 1024 possible 10-tuples are desired, then the parameters \( a, b \) are selected such that

\[
p = \frac{(b-a)}{10,000}
\]

By selecting different values of \( a, b \) such that \( (b-a) \) is constant, then independent historical data bases of approximately the same size can be obtained. If it is desired to have identical size, then random additions or subtractions of each data base are usually necessary. For example suppose a historical data base of size 64 is desired, but the generation process constructs a data set of size 66. Therefore two 10-tuples are randomly selected and deleted.
START

x = 0

Generate a random number, y, from a uniform [0,10000] population

x = x + 1

a \leq y \leq b?

YES

Include binary representation of x in historical data base

NO

x = 1023?

YES

STOP

NO

x = 0

Figure C.1--Generation of Random, Uniform Historical Data Base
APPENDIX D
Listing of Computer Program
The listing of the computer program consists of two parts: the listing of the job control language (JCL) statements, and the listing of the search algorithm. The language is PL/I. The JCL listing is first followed by the algorithm listing.
Table D.1
Listing of Job Control Language Statements

1. Job card
2. /*JOBPARM LINES=1000,REGION=700,TIME=3
3. //EXEC PLIRUN
4. //CMP.SYSIN DD *

: (algorithm source deck)

5. /*
6. //GO.SYSIN DD *
7. //GO.HIST DD DSN=$0968L.HIST.DATA,DISP=SHR
8. //GO.EMPTY DD DCB=(BLKSIZE=12800,LRECL=16,RECFM=FB),
   UNIT=SYSDA,
9. // SPACE=(TRK,(1,1))
10. //GO.PNTNON DD SYSOUT=A,DCB=(BLKSIZE=132,RECFM=FBA,
    LRECL=132)
11. //GO.PNTTEMP DD SYSOUT=A,DCB=(BLKSIZE=132,RECFM=FBA,
    LRECL=132)
12. //GO.SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
13. //GO.SORTIN DD DSN=$0968L.HIST.DATA,DISP=SHR
14. //GO.SORTOUT DD DSN=$0968L.HIST.DATA,DISP=SHR
15. //GO.SORTWKO1 DD UNIT=SYSDA,SPACE=(TRK,(2,2))
16. //GO.SORTWKO2 DD UNIT=SYSDA,SPACE=(TRK,(2,2))
17. //GO.SORTWKO3 DD UNIT=SYSDA,SPACE=(TRK,(2,2))
18. //GO.SYSOUT DD SYSOUT=A
19. //
Table D.2
Listing of Algorithm

SUBSCRIPT-RANGE, STRING-RANGE);   SEARCH: BASIC OPTIONS(MAIN);
/* the scope of the following declarations are for the entire program */

DECLAR
(X, Y, MIN, P, PRIME, XMIN, A, MAX, ») FIXED BIN(15) INIT(0),
X FIXED BIN(15) INIT(3),
(XMIN, 5,6, XMAX), 6, X) BIT(15) INIT(15(15+1))'9,
MAX FIXED BIN(15) INIT(10),
(XG(6, XG_MAX(6), XG_MAX(6)) BIT(15) INIT((6)(15+1))'3,
INDEX BIT(16) VARYING INIT('9),
A(13,10) BIT(1) VARYING INIT((32767)(1))'9,
INDEX BIT(16) VARYING INIT((32767)(1))'9,
COUNT FIXED BIN(16) BASED(P),
COUNT FIXED BIN(16) BASED(P);  
COUNT FILE RECORD.
/* FILE INPUT RECORD:*/

DCL /* DCL REF */

(A(6), TEMP, NULL) BIT(16),
{NUM, COL_NUM, J, COLUMN, ROWNUM, COUNT) FIXED BIN(15);  
/* DCL RESOLUTION */
PUSH (NUM(1, 8) FIXED BIN(15);
/* DCL COMPARABLE */

(DIFF, PROD, KEY_LENGTH) FIXED BIN(15),
KEY BIT(15) VARYING,
KEY BIT(15) VARYING,
LOCAL BIT(15),
SUM KEY(1) BIT(15) VARYING INIT((5)(15+1))'9,
DEF STRING BIT(15) VARYING INIT((15+1))'9,
/* PAINTER,
COUNT FIXED BIN(15);*/
INDEX FIXED BIN(15);  
DCL INDEX FILE FIXED BIN(15) BASED(P);  
/* DCL PRAMABLE */

CYCLES FIXED BIN(15) INIT(0),
PRODUCT BIT(16),
SUM FIXED BIN(15) INIT((6)(15)),
" FAMILY(3,1,3,1,2,1,7,1,2,1,2,1) BIT(1) INIT((5)(1))'9;
Table D.2--continued

DCL /* DCL INITIAL */
   NAME2 FIXED BIN(16),
   H(21A) FIXED BIN(15) INIT(3,1,3,4,4),
   PATCH(21E) BIT(5) VARYING INIT(1(1)11719.2(1)111001A),
   (LMQ,FILL) FIXED BIN(13),
   (B_2,TEMP,B_TEMP2,B_TEMP3) BIT(10) VARYING;
/* DCL INITIAL */
DCL PTR POINTERS;
   PTR=ADDR(TEN);:
DCL AI_TEMP FIXED BIN(15) BASED(PTR);:
DCL /* DCL SORT */
   TIME MULTIN,
   T CHAR(3),
   IA CHAR(13),
   FLO PIC'99',
   #S PIC'99',
   RTCODE FIXED BIN (31),
   0 BIT(14),
   (X8NIEE_-_BAP,CW_SOFT=0) FIXED BIN(15) INIT(0),
   ININ FIXED BIN(15) INIT(1),
   (C1110).C1C00(10) FIXED BIN(15) INIT(11)0),
   DELTA BIT(1) INIT('C'),
   (PT3,PTA) POINTERS;
   PT3=ADDR(X3IN);
   PT3=ADDR(301);
DCL X BIT(16) BASED(PT3),
   O FIXED BIN(15) BASED(PTA),
   1 HISTREC,
   2 RECNUM PIC'Z9999',
   2 RECND PIC'16999',
   2 ROW(10) PIC'9999',
   1 SUMREC,
   2 X_COPY BIT(15),
   2 FILLER CHAR(14):
/** 
* FUNCTION NORM *
*/
/** 
* NORM: PROC(NBIT_STR) RETURNS FIXED BINARY(15)): 
* NORM IS A FUNCTION WHICH TAKES A BIT STRING COUNTS THE 
DCL NBIT_STR BIT(16) VARYING,
   NBIT_STR BIT(16),
   XNT FIXED BIN(15):
   NBIT_STR=NBIT_STR;:
   / SINCE NBIT_STR IS MODIFIED THE ARGUMENT CANNOT BE 
   CO WHILE(NBIT_STR) SPECIFIES THAT THE INSTRUCTIONS CONTAINED 
   IN THE DO AND END ARE TO ARE EXECUTED AS LONG NBIT_STR 
   HAS THE OR MORE '1's IN IT */
   XNT=0:
   DO #WHILE(NBIT_STR):
      XBIT_STR=SUBSTR(NBIT_STR,INDEX(NBIT_STR,1)+1);
      XNT=XNT+1;
Table D.2--continued
Table D.2—continued

THEN XG(prime)=11:
ELSE DO: i=i+1: j=(1:COL_NUM)+1:
GO TO CHECK:
END:

<<<<<<<<<
SUBROUTINE RESOLUTION
<<<<<<<<<

// THIS SUBROUTINE DETERMINES IF THE REP GENERATOR SET,
XG_prime HAS RESOLUTION =3. //

// RESOLUTION: p=3:
SINGLE_CHECK: DO i=1 TO p:
NORM=NORM(XG_prime(I)):
IF NORM<3
THEN GO TO ITERATE1:
END SINGLE_CHECK:

// NEXT RESOLUTION OF ALL POSSIBLE PAIRS OF
XG_prime ARE CHECKED: */
OUTER: DO i=1 TO p-1:
INNER: DO j=(i+1) TO p:
PAIR_SUM=SUM(XG_prime(i),XG_prime(j),1110#*):
NORM=NORM(PAIR_SUM):
IF NORM<3
THEN GO TO ITERATE1:
END INNER:
END OUTER:

// IF ALL OF ABOVE CHECKS SATISFIED THEN
RESOLUTION=3 */
// THEN PROCEED TO COMPARE */

<<<<<<<<<
SUBROUTINE COMPARE
<<<<<<<<<

// COMPARE DETERMINES IF THE CURRENT REP HAS BEEN PRE
VIOUSLY CONSIDERED. IF IT HAS, THEN CONTROL IS PASSED
TO ITERATE1. IF NOT, THEN REP IS STORED IN A LOCATION
BASED UPON ITS HASHING FUNCTION VALUE, AND CONTROL IS
PASSED TO FEASIBLE */

COMPARE: NCOUNT=0: /* INITIALIZE NCOUNT */
// FIRST OBTAIN ARGUMENT FOR HASHING FUNCTION */
SUB_KEY=SUBSTR(XG_prime,17-K+P,K-P):
// SUB_KEY EQUALS LAST K-P BITS OF XG_prime */
KEY=STRING(SUB_KEY):
// NEXT MUST MANIPULATE KEY TO HAVE A LENGTH
// OF 15 BITS /*
KEYLENGTH(K-P)+=:
// NUMBER OF IMPORTANT BITS IN KEY */
IF KEY_LENGTH<15 THEN
STRETCH: DO:
PA=1:KEY_LENGTH:
KEY=REPEAT(10,9,0)|KEY:
LOCATE=SUBSTR(KEY,1,15):
END STRETCH:

IF KEY_LENGTH<15 THEN
SHRINK: DO:
DIFF=(KEY_LENGTH-15)/2:
IF DIFF=0 THEN DIFF=1:

*/
Table D.2—continued

```
LOCAL=101 | SUBSTR(KEY, DIFF, 15);  
END SHRINK;

// THE DIFFERENCE IS MADE UP BY DELETING SOME  
LEADING AND TRAILING BITS */  
IF KEY LENGTH=19  
THEN LOCAL=11 | SUBSTR(KEY, 1, 15);  

// THE LEADING FOR RESULTS IN THE 16 BIT STRING  
HAVING PROPERTIES OF A POSITIVE BINARY NUMBER */  

// NOW READY TO DETERMINE IF XG.ORIG REPRESENTS  
A PREVIOUSLY UNCONSIDERED ALIAS SUBGROUP */  

// ASSUMING URP CASE OF X=10, P=6: THEN X=19; MAX */  

// IF P=6, DELETE LIST 6=0 XG.ORIG IS STRING */  
// XG.ORIG=STRIP(XG.ORIG,17-K,K);  
// DEF_STRING=STRING(KP);  
// DEF_STRING=SUBSTR(DEF_STRING,1,XP);  
// DEF_SET MUST BE INITIALIZED EVERY TIME K AND KP *  
// CHANGE, INITIALIZE TO (15)"*/  
// FIRST_CHECK: IF DEF_SET(BIN_LOCALS)=15 THEN  
// NEW: DEF_STRING=STRING(KP);  
// DEF_SET(BIN_LOCALS)=DEF_STRING;  
// GO TO FEASIBLE:  
// END NEW:  
// SECOND_CHECK: IF DEF_SET(BIN_LOCALS)=DEF_STRING  
// THEN GO TO ITERATE:  
// OLD ALIAS SUBGROUP */  

// IF EITHER FIRST OR SECOND CHECK WAS TRUE,  
// THEN A COLLISION HAS OCCURRED, THEREFORE INCREMENT  
// DEFSUBSCRIPT AND TRY AGAIN */  
// NCOUNT=NCOUNT+1; // NCOUNT IS A COUNTER */  
// */ FOR FINAL VERSION TEST NCOUNT AGAINST 327  
// IF NCOUNT>327 THEN DO:  
// PUT XG.ORIG LIST(1 RESolvable COLLISION,N=327);  
// GO TO STOP_SEARCH:  
// END:  

// ASSUMING ARRAY HAS 32767 LOCATIONS */  
// SIN_Local=0; 
// */ HAS END OF ARRAY BEEN REACHED? */  
// IF SIN_LOCAL=32767 THEN SIN_LOCAL=1;  
// GO TO FIRST_CHECK:  

**********************************************************  
* SUBROUTINE FEASIBLE */  
* **********************************************************  

// FEASIBLE PERFORMS THE FINAL CHECK UPON A ROW ECHOLOG F  
// ORN GENERATORS SET XG.ORIG(1), WHICH HAS "PASSED" ALL OTHER  
// CHECKS. THE FINAL CHECK CONSISTS OF DIVIDING E,THE EMPTY SET  
// INTO THE 200 MEMBERS OF THE FRACTIONAL FACTORIAL FAMILY FOR  
// XG.ORIG. IF AT LEAST ONE MEMBER HAS NO EMPTY CELLS, THEN A  
// FEASIBLE FRAGMENT EXISTS */  
// FEASIBLE: COUNT=0; // INITIALIZE COUNT */  
// FAMILY=(1)"*/  
// CYCLES=CYCLES+1; // CYCLES COUNTS ITERATIONS */  
// IF CYCLES>15 THEN DO:  
// CYCLES=0;
```
Table D.2—continued

```
PUT SKIP DATA((XG(J) DO J=1 TO P));
PUT SKIP:
END:

ON ENDFILE(EMPTY) BEGIN:
A(K,P)='1' B:
CLOSE FILE(EMPTY):
SEC: / * INITIALIZE S */
GO TO SEARCH:
END:

OPEN FILE(EMPTY) INPUT:
NEXT: READ FILE(EMPTY) INTO(DUM_SEC):
X_SUB_Y=X_COPY:
INNER_PRODUCT: DO I=1 TO P:
PRODUCT=XG Điện(I) & X_SUB_Y:
S(I)MOD(NORM(PRODUCT),2):
/* S IS FOR "SUBSCRIPT" */
END INNER_PRODUCT:

/* NOTE: S(I) IS EITHER 0 OR 1 */
IF FAMILY(S(1),S(2),S(3),S(4),S(5),S(6))='0' THEN
SWITCH: DO:
FAMILY(S(1),S(2),S(3),S(4),S(5),S(6))='1' B:
COUNT=COUNT+1
END SWITCH:

/* WHEN COUNT=2**2 THEN EVERY MEMBER OF FAMILY HAS AT LEAST
THE EMPTY CELL */
IF COUNT=2**2 THEN
FAILURE: DO:
CLOSE FILE(EMPTY):
SEC: / * INITIALIZE S */
GO TO ITERATE:
END FAILURE:

GO TO NEXT:
/* IF REACH END OF EMPTY SET AND COUNT<2**2 THEN A FEASIBLE
FRACTION HAS BEEN FOUND */

SUBROUTINE INITIAL:

INITIAL DETERMINES: 1) MAX POSSIBLE RESOLUTION
2) THE MINIMUM NUMBER OF FACTORS REQUIRED FOR RESOLUTION,
3) THE MAXIMUM VALUES FOR GENERATORS, AND
4) THE INITIAL SET OF GENERATORS TO START THE ITERATION

/* INITIAL ASSUMES MAX=10 SUCH THAT : 1) KMAX=11
2) 2*KMAX=20 AND 31*(KMAX(KMAX+1))/2=66 */

INITIAL: K=MAX-1; / * K IS ASSIGNED ITS VALUE HERE */
IF K=1 THEN CASE: DO:
IF K KWIN THEN
CASE_KWIN_CASE: DO:
MAX=K:
XG(1)=139 111 9;
XG_MAX(1) = XG | REPEAT(13,9,<=11):
XG_DOUBLE=XG:
GO TO FEASIBLE:
END CASE_KWIN_CASE:
```
Table D.2—continued

```
ELSE
PI_K_CASE: DO:
    pmax = XGVIN:
    XG(I) = XGVIN(I + 1):
    /* ASSUMING XGVIN IS 16 TUPLE */
    /* BECAUSE DIAGONAL RULE HOLDS FOR D=1, CAN LIMIT XG_MAX(I) TO XGVIN TUPLE IF I'S */
    XG_MAX(I) = REPEAT([I'B, 15-KVIN]) || REPEAT([I'B, KVIN-1])
    XG_IPV = XG:
    GTO FEASIBLE;
END PI_K_CASE;
ELSE XGVIN_CON: DO: /* CODE TO CONSTRUCT XG_MAX */
    LEAN = K-(M(P)-1)+1:
    /* LEAD IS */
    FILL = K-(LEAD)+(M(P)+1): /* FILL EQUALS NUMBER OF O'S PRECEDING PATCH FOR XG_MAX(I) */
    PUT SKIP(9) DATA(K, LEAD, (P), FILL):
    B_TEMP2 = REPEAT(I'B, LEAD-1):
    B_TEMP3 = REPEAT(I'B, FILL-1):
    B_TEMP5 = REPEAT(I'B, K-P-2):
    B_TEMP5 = REPEAT(I'B, D-3):
    IF FILL > XG_MAX(I) = PA0XG||B_TEMP2||B_TEMP1 || PATCH(0):
ELSE
    XG_MAX(I) = PA0XG||B_TEMP2||PATCH(0):
    DO: /* TO P-1 */
    B_TEMP2 = REPEAT(I'B, FILL-1):
    IF FILL > XG_MAX(I) = PA0XG||REPEAT([I'B, 1-2]||B_TEMP2||B_TEMP3 || PATCH(0):
    ELSE
    XG_MAX(I) = PA0XG||REPEAT([I'B, 1-2]||B_TEMP2||B_TEMP3 || PATCH(0):
END;
XG_MAX(P) = XG_MAX(P-1): /* 7/17/77 CHANGE */
    PUT SKIP DATA((XG_MAX(J) DO J=1 TO P)):
    PUT SKIP;
END XGVIN_CON:
GO TO CONSTRUCT:
IF ((I+1) < 2**((K-P)) & (2**((K-P)/2)**(K-P)) THEN
    EMAX_3_CASE: DO:
        EMAX = 3:
        GO TO CONSTRUCT:
END EMAX_3_CASE:
ELSE IF (2**((K-P)/2)**(K-P)) & 2**((K-P)/2)**(1+(K+1)/2)) THEN
    EMAX_4_CASE: DO:
        EMAX = 4:
        GO TO CONSTRUCT:
END EMAX_4_CASE:
/* USING MITCHELL AND DRAPER RESULTS */
ELSE IF 2**((K-P)/2)**(K-P) THEN
    EMAX_5_CASE: DO:
        EMAX = 5:
        GO TO CONSTRUCT:
END EMAX_5_CASE:
ELSE IF 2**((K-P)/2)**(K-P) THEN
  IF (I+1) < 15 THEN
```
Table D.2—continued

GO TO PMAX_A_CASE;
ELSE GO TO PMAX_A_CASE;
ELSE IF 2**(K-Q)=32 THEN
   IF (3<K & K<15) THEN
      GO TO PMAX_A_CASE;
   ELSE GO TO PMAX_A_CASE;
ELSE IF 2**(K-Q)=32 THEN
   IF (3<K & K<15) THEN
      12 TO PMAX_A_CASE;
ELSE GO TO PMAX_A_CASE;
ELSE GO TO PMAX_A_CASE;
ELSE DO: PUT SKIP LIST('ERROR IN SUBROUTINE INITIAL');
   PUT SKIP LIST('K=X,K1=1,P=1');
   PUT SKIP LIST('MAXIMUM RESOLUTION NOT RESOLVED');
END:
/* GIVEN X,P, XMIN(P-1) (1=1 TO P) REPRESENTS THE SMALLEST GENERATOR SET CONSIDERED FOR WHICH A(K,P)=1 *
CONSTRUCT: IF XMIN(P-1)=16!9 THEN SCRATCH: DO: /* MUST CONSTRUCT INITIAL XG(I)*/
   FOM SCRATCH */
   XG(P)=13191119:
   /* XG(P) IS THE SMALLEST GENERATOR */
   OUTER100: I=1 TO (P-1):
   XGMIN(P-1)=XGMIN(P-1)+1
   INNER: NCRI=NCRM(XG(P-1));
   TEM=POOL(XG(P-1),XG(P-1)+1,0110B);
   NORM2=NCRM(TEMP);
   IF (3<NCRI & NCRI<K_P) THEN
      IF 3<NCRI THEN
         GO TO END_CUTER:
      XGMIN(P-1)=XGMIN(P-1)+1
      GO TO INNER:
   END_CUTER: END CUTER:
   I=1; /* MUST INITIALIZE I BEFORE TRANSFER TO ITERATE */
   GO TO CHECK1:
END SCRATCH:
ELSE USE_XGMIN: DO:
   DO I=2 TO P:
      XG(I)=XMIN(P-1)-1):
   END:
   XGMIN(I)=XMIN(2)+1:
   I=1; /* MUST INITIALIZE I BEFORE TRANSFER TO ITERATE */
   GO TO CHECK1:
END USE_XGMIN:

******************************************************************************

SUBROUTINE ITERATE
******************************************************************************

CONTROL IS TRANSFERRED TO ITERATE WHEN 'THE CURRENT' XG(I) HAVE BEEN CONSIDERED AND TO CHECK1
WHEN XG(I) HAVE NOT AS YET BEEN CONSIDERED */
ITERATE:
I=1; /* INITIALIZE */
Table D.2—continued

```plaintext
IF P=1 THEN  
/* THIS SECTION HANDLES THE P=1 CASE */
L1A: DO: IF XG(IN(1))=XG_MAX(IN(1)) THEN  
/* CAN BE EQUAL TO BECAUSE XG(1) HAS  
ALREADY BEEN CONSIDERED */
L2A: DO: A(K,1)=0.00;  
GO TO SEARCH1;
END L2A:
ELSE DO WHILE (XG(IN(1))<XG_MAX(IN(1))):  
XG(IN(1))=XG(IN(1))+1;  
END:
IF A>3 THEN DO:  
XG(IN)=-XG;  
GO TO FEASIBLE;
END:
END;
PUT SKIP LIST(1:ERROR IN LOOP LIA OF ITERATE1!);  
GO TO STOP_SEARCH;
END LIA:
ELSE L3: DO:  
/* P=1 CASE */
L31: DO: WHILE (XG(IN(1))>XG_MAX(IN(1))):  
IF I=P THEN DO: A(I,P)=0.00:  
GO TO SEARCH1;
END:
END:
I=I+1:  
END L31:
XG(IN(1))=XG(IN(1))+1;
END L3:
CHECK1: IF XG(IN(1))>XG_MAX(IN(1)) THEN  
/* HERE "STRICTLY GREATER THAN" IS USED  
BECAUSE XG(I)=XG_MAX(I) IS AN ALLOWED  
VALUE */
IF I<P THEN  
L6A: DO: I=I+1:  
XG(IN)=XG(IN)+1;  
GO TO CHECK1;
END L6A:
ELSE /* THEREFORE */ I=P;  
OCIA(K,P)=0.00;  
GO TO SEARCH1;
END:
/* IF XG(P)>XG_MAX(P) THEN NO FEAS!  
PEL FRACTION EXISTS */
ELSE /* THIS SECTION HANDLES ITERATION FOR NON-XG_MAX CASES */
IF I=1 THEN  
L7: DO:  
L1_CASE: NORM=تهم(1): /* I=1, NON-XG_MAX CASE */  
/* TEMPE=CL(XG(IN),XG(2),1012.3):  
NORM2=Norm(TEMP):  
IF (J>Norm) OR NORM2<k(P) THEN  
IF NORM2>3 THEN  
/* K3 IS AN ARRAY OF MINIMUM ALLOWED K VALUES GIVEN P */  
XG(IN)=XG(IN)+1:  
IF XG(IN)>XG_MAX(IN(1)) THEN GO TO CHECK1;  
ELSE GO TO L1_CASE:
END L7:
ELSE /* THEREFORE */ I>1, NON-XG_MAX CASE */
```

Table D.2—continued

DECREASE_II:
/* THIS SECTION FINDS SUITABLE VALUE FOR XG(I), THEN
DECREASES I */
IF XGMIN(I) < XG_MAXBIN(I) THEN GO TO CHECK I:

NORM2=NORM(XG(I));
IF (XGMIN(I) < XGMAX(I) AND XGMIN(I) < XGMAX(I)) THEN
L93: GO TO L101 IF XG(I) = XGMIN(I);
END L93;

L92: XGMIN(I-1) = XGMIN(I+1);
NORM2=NORM(I-1):
IF NORM2 < XGMIN(I) THEN GO TO L93;
/* THEREFORE XG(I) NOT SATISFACTORY */
IF B1.TEMP > XGMIN(I) THEN GO TO L101;
IF NORM2 > XG THEN GO TO L101;
ELSE GO TO L111 /* THEREFORE PREVIOUSLY CONSIDERED */
END L92;
ELSE IF XG(I) < (P-1) THEN /* THEREFORE CHECK IF NEW ALIAS SUBGROUP */
L91: XGMIN(I-1) = XGMIN(I+1);
NORM2=NORM(I-1):
IF NORM2 < XGMIN(I) THEN GO TO L93;
/* THEREFORE XG(I) NOT SATISFACTORY */
IF B1.TEMP > XGMIN(I) THEN GO TO L101;
ELSE GO TO L111 /* THEREFORE PREVIOUSLY CONSIDERED */
END L91;
ELSE /* INCREMENT XG(I) AND CONSIDER AGAIN */
L111: XGMIN(I) = XGMIN(I+1);
GO TO DECREASE_II;
END L111;

Main Program

START:
/* WHAT IS THE FIRST TIME SORT IS CALLED, CONTROL IS RETURNED */
/* TO SEARCH, ALL SUBSEQUENT TIMES CONTROL IS TRANSFERRED TO SEARCH C */
SEARCH1: /* XMIN */
/* XMIN IS THE SMALLEST NUMBER OF FACTORS FOR WHICH A FULL FACTORIAL DOES NOT EXIST. */
P = 1;
DELTA = DELTA * 3; /* DELTA CHANGES STATE WHEN THE FIRST FEASIBLE CELL IN AN A-MATRIX ROW IS FOUND */
SEARCH2: /* SKP LIST(I) CONTROL TRANSFERRED TO CONTROL2 */
/* SKP DATA(x,k,1,k,x,p) */
/* SKP DATA((XG(I)),XGMIN(I),XG_MAX(I),XG_MAXBIN(I),XG_MAXBIN(I) C J = 1 TO (P)) */
Table D.2—continued

```
IF NEW THEN DONE=TRUE;
GO TO INITIAL; END;
/* NEW=TRUE WHEN A(K,P)=1 AND
XG IS UN-INITIALIZED */
IF A(K,P)=1 THEN
L19: DO: XGTEMP(P,*)=XG;
IF -DELTA THEN
L2: DO: P_MIN=P;
DELTA=1.P;
END L2;
/* P_MIN USED TO DETERMINE COLUMN START
POSITION IN (K+1)TH ROW */
ELSE:
IF P=3 THEN
L4: DO: P.PRIME=P+1;
IF K=5 THEN IF NORM(XG(1))<5 THEN GO TO LSS;
IF K=9 THEN IF NORM(XG(1))>5 THEN GO TO LSS;
DO WHILE ((K+1)<2**(K-P.PRIME)):
A(K,P.PRIME)=1.B;
XGTEMP(P.PRIME,*)=(16)*0'B;
P.PRIME=P+1;
END:
END L4;
ELSE GO TO LSS;
END L3;
ELSE /* THEREFORE 1(K,P)=0'9 */
L5: DO: XGTEMP(P,*)=(16)*0'B;
LSS: DO: P=P+1; /* INCREMENT P */
IF ((K+1)<2**(K-P.PRIME)) THEN DO: NEW='0'B;
ELSE GO TO INITIAL; END;
/ * I.E., CONTINUE SEARCH OF (K)TH ROW */
ELSE /* DELTA THEN DO: */
PUT SKIP LIST(1A(K,P)=1 FOR ENTIRE ROW, K=1,K):
GO TO STOP_SEARCH;
END:
END L5;
END L3;
/* AT THIS POINT THE (K)TH ROW IS COMPLETED AND THEREFORE
IS INCREMENTS. NEXT MUST BE CALLED BEFORE FURTHER
SEARCHING CAN CONTINUE */
XGTEMP=XGMIN:
PUT SKIP DATA((XGMIN(J,KK) DO KK=1 TO J) DO J=1 TO P)):
K=K+1;
PFF_SET=(16)*0'B; /* RESIZING PFF_SET */
IF PMIN=1 THEN P=1;
ELSE PMIN=P+1;
DELTA=1.P;
IF K<KMAX THEN
L5: DO:
PUT SKIP LIST('HAVE EXHAUSTED FACTORS TO CONSIDER');
GO TO STOP_SEARCH;
END L5;
ELSE IF (K+1)>2**(K-P) THEN
L7: DO:
PUT SKIP LIST('FRACTIONAL FACTORIAL NOT CONSTRUCTIBLE');
GO TO STOP_SEARCH;
```
Table D.2—continued

```
SUBROUTINE SORT

C-----------------------------------------------------------------------------
C SUBROUTINE SORT
C-----------------------------------------------------------------------------

C
C TIME:
C
C PUT SKIP EDIT('TIME=',T)(COL(20),A,A);

C FLDO=21-K: /* THEREFORE RECORD LENGTH=21 */
ON ENDFILE(HIST) BEGIN;
CLOSE FILE(HIST);
IF N=MIN THEN DO;
IF K<K THEN
PUT SKIP FILE(PNTPON) EDIT(X,N)(3(15),COL(20),F(5));
END;
ELSE DO;
IF K<K THEN
PUT SKIP FILE(PNTPON) EDIT(X,N)(3(15),COL(20),F(5));
END;
ELSE IF X9IN<2**K-1 THEN DO;
X9IN=X9IN+1;
DO WHILE(X9IN<2**K-1);
END;
END;
CLOSE FILE(PNTPON);
IF X9IN THEN DO;
X9IN=X9IN+1;
END;
ELSE IF X9IN THEN DO;
IF X9IN THEN X9IN=X9IN+1;
END;
ENDDO:

C FILE(IN)

GO TO 00:
END:
C
END:

C DATA

GO TO 00:
END:
```

```

```
Table D.2—continued

```plaintext
OPEN FILE(HIST) INPUT;
OPEN FILE(EMPTY) OUTPUT;
IF < K THEN
  PUT SKIP(5) FILE(PTNONE) EDIT('VOLUME SET';'K=K,K')
                                   (COL(5)*A+COL(20)*A+F(2));
  IF < K THEN
    PUT SKIP(5) FILE(ANTEMP) EDIT('EMPTY SET');(COL(5),I);
  END IF;
END IF;

/* INITIALIZE COUNTERS AND INDICATORS */

XBIN=0;
N=0;
DELTA=1-9;
COL_0=0;
COL_1=0;

3: READ FILE(HIST) INTO (HISTREC);
   /* THEREFORE ROW(16) EQUALS LEVEL OF HIGHEST PRIORITY FACTOR */
   B0=GET(10';15-K)' || SUBSTR(STRING(ROW),11-K,K);
LI: IF DELTA=113 THEN
   LI1A:00: IF XBING=0 THEN
     /* WAS MATCH ON PREVIOUS AND CURRENT RECORD */
     LI1D:00: N=N+1;
     GO TO B;
   ELSE LI1I:
     /* THEREFORE XBIN NE 0 */
     LI12:00: /* WAS MATCH ON PREVIOUS RECORD BUT NOT ON CURRENT
     RECORD */
     N=N+1;
     IF N<=NMIN THEN /* THEREFORE NONEMPTY CELL */
     LI21:00: IF < K THEN
       PUT SKIP(2) FILE(PTNONE)_EDIT(X,N)(B(15),COL(20),F(5));
       S_BAR=S_BAR+1; /* INCREMENT NONEMPTY CELL COUNT */
       COL_1=COL_1+RCW;
       /* ARRAY COL_1 COUNTS TOTAL NUMBER OF 1'S IN EACH COLUMN OF THE NONEMPTY SET */
     END IF;
   ELSE LI22:00: /* THEREFORE EMPTY CELL */
       X_CODE=X;
       WRITE FILE(EMPTY) FROM(DUM_REC);
     IF < K THEN
       PUT SKIP(2) FILE(ANTEMP) EDIT(X,N)(B(15),COL(20),F(5));
       E=0+1; /* INCREMENT EMPTY CELL COUNT */
   END IF;
   END ELSE
   END IF;
   ELSE LI12:
   END IF;
END L11;
```

Table D.2—continued

ELSE L2210: /* THEREFORE DELTA = 1 */
   IF XBIN = 0 THEN
      /* 1: WAS NO MATCH ON PREVIOUS RECORD,
       2: IS MATCH ON CURRENT RECORD,
       3: SINCE THIS IS FIRST MATCH FOR CURRENT
       XBIN = 1. */
      L2210D:
      X = 1;
      DELTA = 1 = 19;
      GO TO 9;
   END L2210;
ELSE /* THEREFORE XBIN = 0 */
   L2220D:
   X_COPY = 1;
   WRITE FILE (EMPTY) FROM (DUM_FEC);
   IF K < 7 THEN
      PUT SKIP FILE (DNUMTEMP) EDIT (X11, COL(2), (5)):
      X BIN = XBIN + 1;
      IF XBIN = 2**K THEN GO:
      * PUT SKIP LIST (ERROR IN SORT; XBIN = 2**K--LOOP L22')
      * GO TO STOP_SEARCH;
     END;
   ELSE GO TO L221;
   END L223;
   CHECKS: COL_MIN = 2**K:
   COL10 = COL - COL1: /* ARRAY ASSIGNMENT STATEMENT */
   DO I = 1 TO (11 - K) BY -1:
      COL_MIN = MIN(COL11, COL2, COL_MIN):
   END: /* THEREFORE COL_MIN EQUA LS MINIMUM NUMBER
   OF 1'S ON 1'S IN ANY COLUMN OF THE
   NONEMPTY SET */
   PL = 1:
   PUT SKIP (2) DATA (COL_MIN);
   IF COL_MIN = 0 THEN GO;
      PUT SKIP (2) LIST (ERROR: COL_MIN = 0 IN SORT);
   GO TO STOP_SEARCH;
   END:
   DO WHILE(COL_MIN < 2***(K - 1 - 1)):
      /* AT LEAST 2***(K - 1 - 1) 1'S AND 1'S MUST BE IN EVERY
      COLUMN, THEREFORE INCREASE Q1 UNTIL THE REQUIRED
      NUMBERS DO NOT EXCEED THE EXISTING NUMBER, COL_MIN */
      IF (K + 1) > (**(K - 1)) THEN GO;
      PUT SKIP LIST (INSUFFICIENT 1'S TO 1'S IN A COLUMN
      FOR EVEN THE SMALLEST CONSTRUCTIBLE FRACTION '1');
      PUT SKIP LIST (Q1, COL_MIN);
   GO TO STOP SEARCH;
   END:
   A(K, P1) = 0:9;
   P1 = P1 + 1:
   END:
   P = MAX (P1, P1): /* P NOW HAS ITS MINIMUM POSSIBLE VALUE */
   PROGRESSAT ('Y', 'S', 15 - K):
   NEW = 1:9;
   IF XMIN wenig GO TO SEARCH2;
   ELSE GO TO SEARCH1;
   STOP_SEARCH; END SEARCH;
APPENDIX E
Tables of Numbers of Unique Subgroups
<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Number of Subgroups of Order 2^2</th>
<th>Minimum Number of Factors in Subgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>35</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>155</td>
<td>145</td>
</tr>
<tr>
<td>6</td>
<td>651</td>
<td>636</td>
</tr>
<tr>
<td>7</td>
<td>2,667</td>
<td>2,646</td>
</tr>
<tr>
<td>8</td>
<td>10,795</td>
<td>10,767</td>
</tr>
<tr>
<td>9</td>
<td>43,435</td>
<td>43,399</td>
</tr>
<tr>
<td>10</td>
<td>174,251</td>
<td>174,206</td>
</tr>
</tbody>
</table>
### Table E.2

Number of Distinct Order $2^3$ Subgroups

<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Number of Subgroups of Order $2^3$</th>
<th>Minimum Number of Factors in Subgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>155</td>
<td>145</td>
</tr>
<tr>
<td>6</td>
<td>1,395</td>
<td>1,375</td>
</tr>
<tr>
<td>7</td>
<td>11,811</td>
<td>11,776</td>
</tr>
<tr>
<td>8</td>
<td>97,155</td>
<td>97,099</td>
</tr>
<tr>
<td>9</td>
<td>788,035</td>
<td>787,951</td>
</tr>
<tr>
<td>10</td>
<td>6,347,715</td>
<td>6,347,595</td>
</tr>
</tbody>
</table>
Table E.3
Number of Distinct Order $2^4$ Subgroups

<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Number of Subgroups of Order $2^4$</th>
<th>Minimum Number of Factors in Subgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>26</td>
</tr>
<tr>
<td>6</td>
<td>651</td>
<td>636</td>
</tr>
<tr>
<td>7</td>
<td>11,811</td>
<td>11,776</td>
</tr>
<tr>
<td>8</td>
<td>200,787</td>
<td>200,717</td>
</tr>
</tbody>
</table>
Table E.4  
Number of Distinct Order 2^5 Subgroups

<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Number of Subgroups of Order 32</th>
<th>Having at Least 6 Factors</th>
<th>Having at Least 7 Factors</th>
<th>Having at Least 8 Factors</th>
<th>Having at Least 9 Factors</th>
<th>Having at Least 10 Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| 5                | 1 | | | | |
| 6                | 63 | 57 | | | |
| 7                | 2,667 | 2,646 | 2,247 | | |
| 8                | 97,155 | 97,099 | 95,503 | 77,527 | | |
| 9                | 3,309,747 | 3,309,621 | 3,304,833 | 3,223,941 | 2,526,198 | |
| 10               | 109,221,651 | 109,221,399 | 109,209,429 | 109,951,759 | 106,463,044 | 81,201,064 |</p>
<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Number of Subgroups of Order $2^6$</th>
<th>Minimum Number of Factors in Subgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>120</td>
</tr>
<tr>
<td>8</td>
<td>10,795</td>
<td>10,767</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9,807</td>
</tr>
<tr>
<td>9</td>
<td>788,035</td>
<td>787,951</td>
</tr>
<tr>
<td></td>
<td></td>
<td>783,631</td>
</tr>
<tr>
<td></td>
<td></td>
<td>695,368</td>
</tr>
<tr>
<td>10</td>
<td>53,743,987</td>
<td>53,743,903</td>
</tr>
<tr>
<td></td>
<td></td>
<td>53,729,503</td>
</tr>
<tr>
<td></td>
<td></td>
<td>53,288,188</td>
</tr>
<tr>
<td></td>
<td></td>
<td>46,334,508</td>
</tr>
</tbody>
</table>
APPENDIX F

Generator Iteration Example
Let $a$, $b$, $c$, $d$, $e$ and $f$ be all possible generator values and in addition

$$a < b < c < d < e < f$$

Then the algorithm would iterate the generator as follows:

- $abc$
- $abd$
- $abe$
- $abf$
- $acd$
- $ace$
- $acf$
- $ade$
- $adf$
- $aef$
- $bcd$
- $bce$
- $bcf$
- $bde$
- $bdf$
- $bef$
- $cdf$
- $cef$
- $def$
BIBLIOGRAPHY


213


Margolin, B.H. "Results on Factorial Designs of Resolution IV for the $2^n$ and $2^n3^m$ Series," Technometrics, 11, 431-444, 1969.


