Determination of Symmetric $VL_1$ Formulas:  
Algorithm and Program SYM4

by

Gerald Martin Jensen

Department of Computer Science  
University of Illinois at Urbana-Champaign  
Urbana, Illinois 61801

December 1975

This work was supported in part by the National Science Foundation under Grant No. NSF DCR 74-03514 and was submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science, 1975.
ACKNOWLEDGEMENT

The author would like to thank all the people who assisted in the development of the current SYM\textsuperscript{b} program and its earlier version. In particular, Professor F. J. Michalski for his helpful suggestions and guidance throughout the development of the SYM\textsuperscript{b} program, Thomas Pan for the initial development of SYM-1, and Richard Chilausky and James Larson for their valuable technical assistance.

The author would like to express his thanks for the partial financial support from NSF grant number DCR 74-03515.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENT</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. THE ALGORITHM</td>
<td>5</td>
</tr>
<tr>
<td>3. USER'S GUIDE FOR SYM</td>
<td>3</td>
</tr>
<tr>
<td>3.1 Input Parameters</td>
<td>3</td>
</tr>
<tr>
<td>3.2 Specification of Storage Parameters</td>
<td>8</td>
</tr>
<tr>
<td>3.3 Specification of Optional Parameters</td>
<td>9</td>
</tr>
<tr>
<td>3.4 Specification of Data Parameters</td>
<td>10</td>
</tr>
<tr>
<td>3.5 Output Parameters</td>
<td>11</td>
</tr>
<tr>
<td>4. CRITERIA FOR THE CHOICE OF THE SYMMETRIC SELECTOR</td>
<td>13</td>
</tr>
<tr>
<td>5. EXAMPLES OF PROGRAM INPUT AND OUTPUT</td>
<td>16</td>
</tr>
<tr>
<td>6. CONCLUSION</td>
<td>27</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>40</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>42</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

This paper contains a description of an algorithm that detects symmetry among variables of a $VL_1$ function, and a user's guide for the program SYM4 which is the PL/1 implementation of the algorithm. If symmetry exists, the program creates a symmetric selector using the symmetric variables. This paper also contains a list of criteria for the selection of symmetric selectors as opposed to non-symmetric selectors, when there is symmetry among the variables of a $VL_1$ function.

A definition of the concept of selectors is given in a previous paper. To make this paper self-contained, however, a brief review of the definition of a $VL_1$ selector is necessary.

A selector is defined as the following form

$$[L \# R]$$

The left part $L$ or referee is a sequence of literals, i.e., variables, $x_i$, or their compliments, $x_i'$. If the sequence has more than one literal, then they are separated by the symbol '+', which denotes normal arithmetic sum. The value of $x_i'$ is $d_i - x_i$, where $d_i$ is the maximum domain element for the $i^{th}$ domain and $x_i$ is the value of $x_i$. The selector relation ($\#$) is one of the following relations $=, \neq, \geq, \leq$.

The right part $R$ or reference is a sequence of one or more non-negative integers in increasing order. They are separated by ';', or ':'. The form $c_1:c_2$ is a compressed way to represent a sequence of consecutive integers from $c_1$ to $c_2$ inclusive. A selector is said to be satisfied if the left part $L$ is in relation $\#$ with the right part $R$. The
following is an example of a selector:

\[ [x_1 \neq 1, 3, 5, 7] \]

This selector is satisfied if the value of \( x_1 \) does not equal 1, 3, 5, 6 or 7.

A **symmetric selector** is a selector in which there is more than one literal in the referee (L). The following is an example of a symmetric selector:

\[ [x_1 + x_2 + x_5 = 0, 5, 7] \]

Domains of Input Variables

\[ D_1 = \{0, 1, 2, 3, 4, 5\} \]

\[ 1 = 1, 2, 5 \]

The above formula is satisfied if the variables \( x_1, x_2 \) and \( x_5 \) arithmetically sum to 0 or 5 or 7. The function expressed by the above formula is symmetric with regard to variables \( \{x_1, x_2, x_5\} \) since the value of these variables can be exchanged one for another without changing the value of the function, i.e., whether it is satisfied or not.

The following definition of terms is helpful in the understanding of the algorithm presented in the next section. A **D_Group** is defined as a group of variables with the same number of levels. If a set of variables is symmetric, the values of particular symmetric variables can be exchanged for the value of another symmetric variable. The sum of the symmetric variables would still be the same. If the symmetric variables did not have the same number of levels, exchange of values of two symmetric variables might result in one variable having a value outside its domain. There would be no symmetry in that case. Symmetric variables, therefore, have the same number of levels and are members of the same D_Group.
A completely specified symmetry is defined as a symmetry in which the event class contains all possible ways for the symmetric variables to sum to a particular number. If this condition is not met, the symmetry is incompletely specified.

The Column_Sum for a variable $x_i$ is defined as the sum of the $i$th component of events in one event class. The complement of a Column_Sum is defined as the sum of the complements of $i$th components of events in one class. The compliment of a Column_Sum is also calculated by subtracting the given Column_Sum from the product of the number of events in the class and the largest domain element, (i.e., $\# \text{ events} \times \mathbb{d}_i - \text{Col-Sum}_i = \text{Comp. Col-Sum}_i$). A CS_Group is defined as a group of variables with the same Column_Sum.

If symmetry is completely specified, the symmetric variables will have the same Column_Sum, unless a symmetric variable is complimentary. If the symmetric variable is complimentary, the compliment of its Column_Sum will equal the Column_Sum of the other symmetric variables. If the symmetry is incompletely specified, nothing can be said about the relationship between the Column_Sums of the symmetric variables.

The Row_Sum for a set of variables is defined as the sum of values of the variables. If the variable $x_i$ is complemented, its value is the complimentary value of the variable $\mathbb{d}_i - x_i$.

For a group of variables with the same domain, total
symmetry is defined as symmetry with regard to all variables of that group. Partial symmetry is defined as symmetry with regard to some, but not all variables of the group. A minimal symmetric selector is a symmetric selector which covers the given event set but has a minimum number of references, and therefore a minimal symmetric covering.

The Class Compact Ratio is defined as the ratio of the number of events in a given class to the total number of events covered by the selector(s) in the formula for the given class. The Minimum D_Group Ratio is defined on an event space of reduced dimension. The dimension of this reduced event space is equal to the number of variables in the D_Group. The D_Group Ratio is the Class Compact Ratio of this event space of reduced dimension. Taking the minimum of these values over all D_Groups in the given class formula, gives the minimum D_Group Ratio.

To compute the number of ways $k$ variables with the same maximum value ($\tilde{x}$) can sum to $R$ (Row_Sum) the following is computed. First choose the minimum of the Row_Sum or Maximum Row_Sum - Row_Sum, which will be denoted Row_Sum*. Determine how many times Row_Sum* is integrally divisible by the number of levels of the variables (i.e., $\omega = \left\lfloor \frac{\text{Row}_\text{Sum}^*}{\# \text{of Levels}} \right\rfloor$). The number of ways $k$ variables sum to $R$ is then

$$\sum_{k_1} (-1)^{k_1} \binom{\omega}{k_1} \binom{\omega + (k_1 - 1)}{k_1}$$

where $k_1$ is initially the value of Row_Sum* and is decreased by the $\#$ of levels for each term in the sum ($k_1 = \text{Row}_\text{Sum}^* - 1^\# \text{of Lev}$).
2. THE ALGORITHM

The following is a description of the algorithm used to detect symmetry in variable-valued Logic Functions. The algorithm operates on user-specified disjoint event classes.

1. Check for disjoint event classes. Intersecting event classes are not considered in this algorithm.

2. Partition all variables into D_Groups. (i.e., variables with the same cardinality). If the user specifies a particular subset of the given variables, (SPECLIST), only those variables specified will be partitioned into D_Groups.

For each Event Class:

3. Ignore D_Groups containing only one variable.

4. Compliment appropriate variables in each D_Group using one of the two compliment routines.

A. Default Routine: If symmetry is completely specified, the Column_Sums of two Complimentary symmetric variables will be complimentary. A check of the Column_Sums determines which variables should be complimented.

B. Optional Heuristic Routine: If the symmetry is incompletely specified, there is no relation between Column_Sums to indicate which variables should be complimented. The Heuristic Routine compliments different variables, then collects and counts the number of Row_Sums generated by this particular compliment vector. Different vectors are tried until a single Row_Sum is generated or all ways of complimenting the variables are exhausted. In the latter case, the compliment vector producing the minimum number of Row_Sums is used.
5. Check all variables in each D_Group for total symmetry. (See symmetry test below) If the test succeeds, the variables in the D_Group are linked to the output list of symmetric variables.

6. For each D_Group that Step 5 fails, further partition into CS_Groups and sort the CS_Groups in descending order based on the number of variables in the particular CS_Group.

7. Ignore CS_Groups containing only one variable.

8. Check all variables in each CS_Group for total symmetry with respect to this group. If the test succeeds, link variables in the CS_Group to the output list of symmetric variables.

9. For each CS_Group that Step 8 fails, find partial symmetry of variables in each CS_Group using pairwise symmetry test. If the test succeeds, link pairs of variables to the output list of variables.

10. If the data reduction switch is specified by the user, collect lists of symmetric variables common to this class and all previous classes.

11. If rerun is specified and no symmetric variables were found in any of the classes, the algorithm is restarted at Step 2 using all the variables. The algorithm is rerun only once.

12. If the data reduction switch is specified, the symmetric variables common to all classes are displayed in a table showing the reduction.

End of Algorithm

The total symmetry test and the pairwise symmetry test are the same symmetry test applied to different sets of variables. The total symmetry for a D_Group or CS_Group is the symmetry test applied to all the variables of the respective group. Pairwise symmetry is defined as the symmetry test applied to exactly two variables.
The symmetry test is satisfied:

1. If the events in other event classes do not sum to the same number as the events in the given event class.

   **AND**

   2a. If all possible ways for the symmetric variables to sum to a particular number are contained in the event class.

   **OR**

   2b. If the ratio of number of events in this event class to the total number of events satisfying condition 2a is greater than a user specified compact ratio (i.e., the selector is compact).

The following is an example of the symmetry test.

If one event in an event class has the property $x_1 + x_2 + x_3 = 3$, then to specify the symmetric selector ($x_1 + x_2 + x_3 = 3$) requires that events in all other event classes do not satisfy the condition $x_1 + x_2 + x_3 = 3$, and all other possible ways to sum to 3, with these variables, must be present in this event class.

**OR**

\[
\frac{\#\text{events in this event class that sum to } 3}{\#\text{of possible ways for these } 3 \text{ variables to sum to } 3} \leq \text{User specified compact ratio (typically 1.0 or 0.5)}
\]

**AND**

Events in all other event classes do not satisfy the condition $x_1 + x_2 + x_3 = 3$. 
3. USER'S GUIDE FOR SYL

3.1 Input Parameters

The use of the program requires the proper specification of the input parameters. The following is a description of the input parameters and their formats. The description of each parameter is as follows:

i. Name of parameter and an example of specification. The value in the example will be the default value for parameters with default specifications.

ii. A list of possible values of the parameter.

iii. A description of the parameter's meaning.

The input parameters can be divided into three groups:

A. Storage Specification Parameters (specified in FL/1 list format) which provide necessary information about array sizes in the program. These parameters have no default values and must be presented in the order specified.

B. Optional Parameters (specified in FL/1 data format) which specify the options that should be used during program execution. Since these parameters are specified in FL/1 data format, the last parameter must be followed by a semicolon. All optional parameters have default values and if a parameter is not specified by the user in his input data, the default is assumed.

C. Data Parameters (specified in FL/1 list format) which specify the $V_L$ functions which are checked for symmetry. Since these parameters are also in FL/1 list format, they have no default values and must be presented in the order specified.
3.2 Specification of Storage Parameters

- NV
  
  Example: 7
  
  Possible Values: Any positive integer (smaller than $2^{15}$)
  
  This parameter specifies the number of variables used in the description of the input events.

- NROW
  
  Example: 16
  
  Possible Values: Any positive integer
  
  This parameter specifies the total number of events in all the event classes. This parameter together with ($NV$) gives the size of the event array.

- MAXCL
  
  Example: 1
  
  Possible Values: Any positive integer ≠ NROW
  
  This parameter specifies the number of the last class of events. The program automatically assigns the number 0 to the first class of events, 1 to the second class and MAXCL to the last class of events. The number of classes of events is MAXCL + 1. This parameter is used in specifying arrays containing class boundary information.

- MAXD
  
  Example: 3
  
  Possible Values: Any possible integer.
  
  This parameter specifies the largest domain element from the domains of all the variables. It is used in the specification of arrays containing the number of occurrences of particular Row_Sums. This parameter with NV gives a ceiling on the maximum Row_Sum. (i.e., $NV^*MAXD$)
3.3 Specification of Optional Parameters

- **HAXPAT**
  
  **Example:**  
  
  `HAXPAT=500`
  
  **Possible Values:** Any positive integer
  
  This parameter specifies the maximum size of the symmetry pattern (i.e., maximum number of events a particular selector can cover).

- **GET**
  
  **Example:**  
  
  `GET='0'3`
  
  **Possible Values:** '0'3, '1'3
  
  If GET='1'3, a heuristic routine is used to decide which variables should be complemented. The heuristic used in as follows: the variables in the D_Group should be complemented to achieve a minimum number of different Column_Sum (ideally one).

  If GET='0'3, a variable is complemented if its complimentary Column_Sum is equal to the Column_Sum of another variable(s) in the D_Group.

  The heuristic Routine is used when some of the events in the symmetry are not specified. In this case the Column_Sum of a variable need not equal the complimentary Column_Sum of another variable even though the first variable and the second variable are symmetric.

- **USER5**
  
  **Example:**  
  
  `USER5SPEC=0`
  
  **Possible Values:** Any positive integer ≠ 0
  
  This parameter allows the user to specify how many of the variables should be checked for symmetry. If 0, (the default) is specified, the program checks all the variables for symmetry.

- **PERM**
  
  **Example:**  
  
  `PERM='0'3`
  
  **Possible Values:** '0'3, '1'3
This parameter is used in conjunction with the USESPEC parameter. If the program fails to find symmetry between the user specified variables and RERUN='1'B, then the program will rerun the problem checking for symmetry between all the variables.

**DTRED**

Example: \[ DTRED='0'B \]

Possible Values: '0'B, '1'B

If DTRED='1'B, the selectors which have identical referees and are all common to all classes are saved. A table is then printed containing the old variables plus new variables defined by the saved selectors. The value of the new variable for a particular event is the value of the reference of its defining selector for that event.

**3.3.1 Type Semi-colon (;)**

Specification of Optional Parameters should be followed by a semi-colon since PL/1 Data Format is used. The semi-colon should be typed even if no options are specified.

**3.4 Specification of Data Parameters**

**• CRITERIA1**

Example: \[ 0.5 \]

Possible Values: Any positive real number \( \leq 1.00 \)

The parameter is the minimum D_Group compact ratio specified by the user. Compact ratio is defined as the ratio of the number of events per class to the number of events covered by a set of symmetric variables. If the compact ratio for a set of symmetric variables is less than CRITERIA1, the set of variables is not used as a selector.

**• DLIST (NV)**

Example: \[ 4,4,4,3,3 \]

Possible Values: A list of NV positive integers with each value \( \leq \text{MAXD} + 1 \)
This parameter specifies the number of levels (or possible values) \( \text{DIST}(i) \) for each variable. Since the minimum value of a variable is zero, the number of levels for a variable \( x_i \) is equal to one larger than the maximum value of that variable. In the above example, the first three variables have four possible values 0, 1, 2, 3, while the last two variables have three possible values 0, 1, 2 (i.e., four levels and three levels respectively).

\[ \text{NE}(0:\text{AXCL}) \]

Example: \[ 10,6 \]

Possible Values: A list of \( \text{AXCL} + 1 \) positive integers which sum to the value of \( \text{NROW} \).

This parameter specifies the number of events in each class. In this example, Class 0 has ten events and Class 1 has six events.

\[ \text{E}(0:\text{ROW}) \]

Example:

\[
\begin{array}{ccccccc}
10&1&2&1&3&1&1
0&0&0&1&3&1
3&1&0&3&2&0
1&1&3&0&1&2
2&3&0&1&0&1
1&1&3&0&1&2
1&1&3&0&1&2
2&1&2&0&3&0
3&2&1&0&2&0
3&2&2&0&0&1
0&0&2&5&2&2
0&1&1&2&2&0
1&1&2&3&3&2
1&1&2&3&3&2
2&2&1&3&0&1
2&2&0&2&2&2
\end{array}
\]

Possible Values: A set of non-negative integers which represent \( \text{NROW} \) events of \( \text{NV} \) variables.

\( E \) is a list of events defining the given \( \text{VL}_i \) function. There are no default values and each value corresponding to the value of variable \( x_i \) must be less than \( \text{DIST}(i) \).

The following parameter is specified if \( \text{USER3=EC4O} \). It is specified in PL/1 list format.
3.3. Output Parameters

The initial portion of the output is a repetition of the input parameters. The following parameters are program generated. The description format of the input parameters is the same as the format of the input parameters. The actual printed output is in tabular form with one class per line, but the following examples show only the column head and the value of the particular parameter.

3.5. FORMULAS

Example:

```
CLASS FORMULAI
F(0) := (x_2 + x_3 + x_7 + 3.5) \& (x_1 + x_4 + x_5 + x_6 + 10)
F(1) := (x_2 + x_3 + x_7 + 4) \& (x_1 + x_4 + x_5 + x_6 + 0.11.12)
```

This parameter specifies the symmetric selectors that cover the events of a class. Selectors are separated by an
"&" sign indicating logical conjunction. A quote mark following a variable specification is taken to mean the compliment of a variable.

- **CLASS COMPACT RATIO**

Example:

```
CLASS
COMPACT
RATIO
0.100
0.167
```

This parameter specifies the ratio of the number of events of a given class to total number of events covered by the selector(s) of the given class.

- **MIN D_GROUP RATIO**

Example:

```
MIN
D_GROUP
RATIO
1.000
1.000
```

The compact ratio, which is the ratio of number of events specified to number of events covered, is generated for all D_Groups (selector variables in output). The minimum of these values over all D_Groups is the value specified by this parameter.

- **NUMBER OF SYMMETRIC VARIABLES PER CLASS**

Example:

```
NUM. OF
SYM VARS
PER CLASS
7
7
```

This parameter specified the number of variables in the symmetric selector(s) for a particular class.
THE DATA REDUCTION TABLE

Example:

| VARIABLE X8 IS THE SUM OF VARIABLES X2, X3, X7 |
| VARIABLE X9 IS THE SUM OF VARIABLES X1, X4, X5, X6 |

<table>
<thead>
<tr>
<th>CLASS FE (0)</th>
<th>EVENT NO.</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>X9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLASS FE (1)</th>
<th>EVENT NO.</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>X9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0</td>
<td>7</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

This table illustrates how a group of symmetric variables, common to all classes, can be used to define a new variable. The new variable is used to describe the classes in a more compact way, and reduces the number of variables that need inspection for determining class membership.
4. CRITERIA FOR THE CHOICE OF THE SYMMETRIC SELECTOR

Currently, there exists a powerful program* AQVAL/1 version AQ7 which synthesizes quasi-optimal formulas for variable-valued logic functions using non-symmetric selectors. What are the advantages of the symmetric selectors generated by the SYM4 program as compared with the non-symmetric selectors generated by the AQVAL/1 (AQ7) program?

If the given variable-valued logic function contains symmetric variables, the advantages are two-fold.

1. A reduction of the time needed to evaluate the selectors to determine class membership.

2. A reduction of the space needed to store the formula.

The following two examples (Fig. 1-4) illustrate these advantages. The control parameters for the AQVAL portion of the examples were chosen to closely parallel the type of covers the SYM4 program generates. (i.e., disjoint covers with no restrictions on the selector references.) The data parameters were the same for both programs. The formulas in the boxes are the output of the respective programs.

To evaluate the selector generated by the AQVAL program, each referee must be compared with each reference. This requires the execution of a compare instruction and branch instruction. To evaluate a symmetric selector, the variables must first be summed, then the sum is compared with each reference. This again requires the compare and branch

* A program description and user's guide for AQVAL/1 (AQ7) is found in paper².
instructions. Furthermore, if a variable is complimented, a subtraction is required to obtain the complimented value of the variable.

The formulas generated by the AQVAL program consist of the logical disjunction of complexes of selectors. All the selectors of a complex must be satisfied for the complex to be satisfied, but only one complex need be satisfied for the formula to be satisfied. To be conservative, an event can satisfy a class formula if it satisfies the first complex of the formula, but if an event does not satisfy a class formula, all the complexes must be evaluated. It will be assumed, therefore, that on the average, one half of the complexes must be evaluated for each class. The formulas generated by the SYM4 program use logical conjunction and therefore, all the selectors must be evaluated to determine if the class formula is satisfied.

The evaluation times in the following table were computed using the instruction times for an I.B.M. 360 Model 75 I,J found in reference3, and using the above assumptions on the evaluation of formulas. The instruction times used are as follows:

\[
\begin{align*}
\text{Add time} & = 0.68 \mu\text{sec.} \\
\text{Subtract time} & = 0.68 " \\
\text{Compare time} & = 0.68 " \\
\text{Branch time} & = 1.01 \\
\end{align*}
\]
Time Required to Evaluate Formulas
(All times in microseconds)

<table>
<thead>
<tr>
<th></th>
<th>Class 0</th>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
<td>AQVAL</td>
<td>13.52</td>
<td>28.73</td>
<td>27.04</td>
</tr>
<tr>
<td></td>
<td>SYM4</td>
<td>5.42</td>
<td>3.73</td>
<td>3.73</td>
</tr>
<tr>
<td>Example 2</td>
<td>AQVAL</td>
<td>18.59</td>
<td>27.04</td>
<td>8.47</td>
</tr>
</tbody>
</table>

The evaluation of the SYM4 formulas for these examples is clearly faster.

The AQVAL program stores complexes as bit strings. The AQVAL program stores as complex as bit strings. The length of the bit string is equal to the total number of levels of all the variables. Since the bit strings are aligned, the storage required is \( \text{MOD}_9(\text{Total} \# \text{Levels}) \) bytes. The SYM4 program stores its formulas as a list of variable subscripts, a list of references and a complement vector for all the variables. Each item in the lists requires one byte per term. These storage assumptions give the following table of storage requirements.

<table>
<thead>
<tr>
<th>Storage Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>(in bytes)</td>
</tr>
<tr>
<td>Class 0</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>Example 1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Example 2</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

In most cases, the SYM4 formulas required less storage.

*For further information—see paper2
Another advantage of the symmetric selectors is in the area of learning by inductive inferences. The formulas generated by AQVAL which describe the event classes, are based upon the specific value of particular variable. The formulas generated by SYM4 which describe the event classes, are based upon the relationship (arithmetic sum) between particular variables. If the given input information contains an incompletely specified symmetry, the symmetric selector covering it represents a generalization of that information. This generalization is considered machine learning in paper. Both programs generate these generalized coverings of events. In a system which represents knowledge as these generalizations, both programs could be employed: one to represent class distinctions based on values of the characteristics (variables) of a class, and the other to represent class distinctions based on the relationship between the characteristics.

If a variable-valued logic function contains symmetric variables, it is advantageous to represent these variables with symmetric selectors generated by the SYM4 program as opposed to non-symmetric selectors generated by AQVAL.
GENERALIZE IS ON
USERSPEC IS OFF
PERMIT IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE = 100
CONTACT RADIUS CRIT = 0.25
NUMBER OF VARIABLES = 3
NUMBER OF EVENTS = 35
NUMBER OF CLASSES = 4
MAXIMUM LEVEL = 3

NUMBER OF LEVELS FOR EACH VARIABLE: 4 4 4
NUMBER OF EVENT SPECIFIED FOR EACH CLASS:

CLASS EVENTS
0 0 6
1 1 12
2 2 11
3 3 6

CLASS F1(0)
EVENT NO. 1 = 1 3 0
EVENT NO. 2 = 0 2 0
EVENT NO. 3 = 0 3 1
EVENT NO. 4 = 2 3 0
EVENT NO. 5 = 1 2 0
EVENT NO. 6 = 0 1 0

CLASS F1(1)
EVENT NO. 1 = 1 3 0
EVENT NO. 2 = 2 1 0
EVENT NO. 3 = 1 0 0
EVENT NO. 4 = 3 1 1
EVENT NO. 5 = 2 1 1
EVENT NO. 6 = 1 1 1
EVENT NO. 7 = 0 0 1
EVENT NO. 8 = 2 3 2
EVENT NO. 9 = 1 2 2
EVENT NO. 10 = 0 1 2
EVENT NO. 11 = 1 3 3
EVENT NO. 12 = 0 2 3

CLASS F1(2)
EVENT NO. 1 = 1 3 0
EVENT NO. 2 = 2 0 0
EVENT NO. 3 = 3 2 1
EVENT NO. 4 = 2 1 1
EVENT NO. 5 = 3 3 2
EVENT NO. 6 = 2 2 2
EVENT NO. 7 = 1 1 2
EVENT NO. 8 = 0 0 2
EVENT NO. 9 = 2 3 3
EVENT NO. 10 = 1 2 3
EVENT NO. 11 = 0 1 3

CLASS F1(3)
EVENT NO. 1 = 1 3 0
EVENT NO. 2 = 3 1 1
EVENT NO. 3 = 2 0 1
EVENT NO. 4 = 2 2 3
EVENT NO. 5 = 1 1 3
EVENT NO. 6 = 0 0 3

CLASS FORMULI

\[ F(0) := (X_1+X_2+X_3=1,2) \]
\[ F(1) := (X_1+X_2+X_3=4) \]
\[ F(2) := (X_1+X_2+X_3=3) \]
\[ F(3) := (X_1+X_2+X_3=6) \]

SYSLIN Output for Example 1 Section 4

Fig. 1

<table>
<thead>
<tr>
<th>CLASS</th>
<th>MIN GROUP RATIO</th>
<th>NUM. OF SYM VARS PER CLASS</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1(0)</td>
<td>0.567</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>F1(1)</td>
<td>1.000</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>F1(2)</td>
<td>0.917</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>F1(3)</td>
<td>0.600</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
space allocated for core is 200
space allocated for m is 75
maximum star size with tracing will be 153 then it will be cut to 50 when oc
log trace 0 7 star trace 0 9 quick list trace 0 9 quick star trace 0 9 save cover data 0 9 save log 0
input format is vector
all variables will be divided by factors
number of variables 1 3
number of levels for each variable: 4 4 4
number of events specified for each class:
class events
0  4
1  12
2  11
3  6
** amount of unused core = 04**

clist= 0 1 2 3
ncr= 2
cist= 1 2
tlist= 0.00 3.00
log-star option lost= 119

class fe 01
event no. 1 2 3 0

class fe 07
event no. 1 2 3 0

class fe 11
event no. 1 2 3 0

class fe 15
event no. 1 2 3 0

aqval/1 (aq7) output for
example 1 section 4

fig. 2

note: figure 2 is continued on
the following two pages.
The following 1 Cartesian complexes cover the events in class 1:

<table>
<thead>
<tr>
<th>Complex</th>
<th>Cov New</th>
<th>Ind</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x1 1 1)</td>
<td>1 1 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x2 1 1)</td>
<td>1 1 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x3 1 1)</td>
<td>1 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x1 0 1)</td>
<td>0 1 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x2 0 1)</td>
<td>0 1 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x3 0 1)</td>
<td>0 0 1 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Delta for this set is 0

The largest star had 5 elements; the largest intermediate star had 9 elements.

The following 1 Cartesian complexes cover the events in class 2:

<table>
<thead>
<tr>
<th>Complex</th>
<th>Cov New</th>
<th>Ind</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x1 2 1)</td>
<td>2 2 2 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x2 2 1)</td>
<td>2 2 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x3 2 1)</td>
<td>2 1 2 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Delta for this set is 1

The largest star had 3 elements; the largest intermediate star had 5 elements.

Figure 2 (cont'd)
### AQVAL/1 (AQ7) Output for Example 1 Section 4

Fig. 2 (continued)

<table>
<thead>
<tr>
<th>COMPLEX:</th>
<th>(x1, 3)</th>
<th>(x2+ 3)</th>
<th>(x3* 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPLEX:</td>
<td>(x1* 3)</td>
<td>(x2+ 1)</td>
<td>(x3* 1)</td>
</tr>
<tr>
<td>COMPLEX:</td>
<td>(x1* 2)</td>
<td>(x2+ 1)</td>
<td>(x3* 1)</td>
</tr>
<tr>
<td>COMPLEX:</td>
<td>(x1* 2)</td>
<td>(x2+ 2)</td>
<td>(x3* 3)</td>
</tr>
<tr>
<td>COMPLEX:</td>
<td>(x1* 1)</td>
<td>(x2+ 1)</td>
<td>(x3* 3)</td>
</tr>
<tr>
<td>COMPLEX:</td>
<td>(x2* 0)</td>
<td>(x3* 3)</td>
<td></td>
</tr>
</tbody>
</table>

**** DELTA FOR THIS SET IS 0 ****

**THE LARGEST STAR HAD 2 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 3 ELEMENTS**

*** NORMAL TRANSFORMATION ***
**SYM4 Output for Example 2 Section 4**

*Fig. 3*

<table>
<thead>
<tr>
<th>CLASS FORMULA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F'(0) = (X2 X3 X7 3, 5) &amp; (X1 X4 X5 X6 = 10)$</td>
</tr>
<tr>
<td>$F'(1) = (X2 X3 X7 4) &amp; (X1 X5 X6 = 0) [11, 12]$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLASS</th>
<th>MIN</th>
<th>NUM. OF SYM VARS</th>
<th>COMPACT RATIO</th>
<th>D GROUP RATIO</th>
<th>PER CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10</td>
<td>1.000</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.167</td>
<td>1.000</td>
<td>7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**DISJOINT COVERS**

**DEFAULT SPECIFICATIONS**

**FACTOR VARS**

Space allocated for gift is 220

Space allocated for IQ is 25

Maximum star size before trimming will be done = 150 then it will be cut to 50 mode=DC

LQ Trace = 0  Star Trace = 0  Quick LQ Trace = 0  Quick Star Trace = 0  Save Cover Data = 0  Save LQ = 0

Input format is vector

All variables will be covered by factors

Number of Variables = 7

Number of levels for each variable: 4 3 3 4 4 3

Number of events specified for each class:

<table>
<thead>
<tr>
<th>Class</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

**Amount of unused core = 8K **

<table>
<thead>
<tr>
<th>OLIST</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORT</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CLIST</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>TLIST</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

LQ-STAR OPTION LOST = **' '**

**CLASS 1: 0**

<table>
<thead>
<tr>
<th>Event No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event No.</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Event No.</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>9</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

**CLASS 1: 1**

<table>
<thead>
<tr>
<th>Event No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event No.</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Event No.</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

AQVAL/1 (AQ7) Output for Example 2 Section 4

Fig. 4
THE FOLLOWING 4 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 0

<table>
<thead>
<tr>
<th>COMPLEX</th>
<th>EVENTS</th>
<th>COVER</th>
<th>NEW</th>
<th>IND</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(X1= 1, 3) (X2= 0, 1) (X4= 1, 3) (X5= 1, 3)</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>(X1= 2) (X2= 1, 2)</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>(X3= 1, 2) (X6= 2)</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>(X3= 2) (X4= 2)</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

**** DELTA FOR THIS SET IS 1 ****

THE LARGEST STAR HAD 53 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 51 ELEMENTS

THE FOLLOWING 5 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 1

<table>
<thead>
<tr>
<th>COMPLEX</th>
<th>EVENTS</th>
<th>COVER</th>
<th>NEW</th>
<th>IND</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(X1= 2) (X2= 0) (X4= 1) (X6= 3)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>(X1= 1) (X3= 1) (X4= 2) (X6= 3)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>(X1= 2) (X4= 0, 3) (X5= 0, 2) (X6= 0, 3)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>(X1= 3) (X2= 2) (X3= 0)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>(X1= 3) (X2= 2) (X3= 1) (X6= 3)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

**** DELTA FOR THIS SET IS 0 ****

THE LARGEST STAR HAD 2 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 2 ELEMENTS

*** NORMAL TERMINATION ***

AQUAL/1 (AQ7) Output for Example 2 Section 4

Fig. 4 (continued)
5. EXAMPLES OF PROGRAM INPUT AND OUTPUT

Several examples of the input and output for the program SYM4 are presented below. They illustrate the modification of the output due to the specification of the various optional routine parameters. In addition, the first example is illustrated with GLD representations of the formulas generated by the program and the input example is in tabular form with the value of the variable, name of the variable, and brief description of the variable on each line.

The execution of the SYM4 program requires the specification of the ID parameters and JCL parameters before the specification of the input parameters. The ID parameters contain time and region estimates. The value of these two parameters is related to the size of the event set used in the problem and the specification of the 'GEN' parameter. Samples of various problem sizes and the amount of time and region needed for successful execution of the program are given in figure 5. The program consists of 580 PL/1 source statements and the object module is 100k bytes long, including the PL/1 library routines. In general, a small problem of several variables requires a region of 116k and a time estimate of 10 seconds or less. For a very large problem, a region of 200k and a time estimate of ten minutes should be sufficient. The heuristic routine selected with the specification of the optional parameter
### REGION AND TIME ESTIMATES

<table>
<thead>
<tr>
<th>No. of Variables</th>
<th>Total No. of Events</th>
<th>No. of Classes</th>
<th>Total No. of Levels</th>
<th>'GEN'</th>
<th>Time</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>35</td>
<td>4</td>
<td>12</td>
<td>No</td>
<td>(0, 0.60)</td>
<td>161k</td>
</tr>
<tr>
<td>3</td>
<td>35</td>
<td>4</td>
<td>12</td>
<td>Yes</td>
<td>(0, 0.62)</td>
<td>161k</td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>2</td>
<td>21</td>
<td>No</td>
<td>(0, 0.42)</td>
<td>161k</td>
</tr>
<tr>
<td>13</td>
<td>16</td>
<td>2</td>
<td>48</td>
<td>No</td>
<td>(0, 0.75)</td>
<td>161k</td>
</tr>
<tr>
<td>35</td>
<td>260</td>
<td>15</td>
<td>100</td>
<td>No</td>
<td>(04, 00)</td>
<td>200k</td>
</tr>
<tr>
<td>35</td>
<td>260</td>
<td>15</td>
<td>100</td>
<td>Yes</td>
<td>(30, 00)</td>
<td>200k</td>
</tr>
<tr>
<td>50</td>
<td>287</td>
<td>3</td>
<td>130</td>
<td>No</td>
<td>(24, 00)</td>
<td>250k</td>
</tr>
</tbody>
</table>

*Figure 5*
'GEN' requires considerably more time in problems with large numbers of variables. This effect should be noted in the specification of a time estimate for problems of this type.

The JCL parameters follow the ID parameters. These parameters are required to gain access to the program. Following the JCL parameters are the input parameters for each problem.

The first example demonstrates the effect of specifying the optional parameter 'GEN'. Generalized logical diagrams (figures 6 and 7) are used to illustrate the formulas generated by the program. Note that the two latter classes are not covered when 'GEN' is not specified.

The GLD is a graphical model of the event space and complexes covering specific events. A complete description of GLDs can be found in paper⁵. Each cell of a GLD represents an event in the event space. The area(s) enclosed by curved lines represent events covered by selectors. In these particular GLDs, the variable $x_1$ specifies the row in which an event is placed, and variable $x_2$ and $x_3$ specify the column.

The numbered cells represent the given input events describing the $VL_1$ function. The particular number(i) in the cell indicates that the event was a member of the $i^{th}$ input class. The variable $x_2$ is complimented and therefore the complimented value of that variable, shown below the non-complimented value, is used to determine if an event is covered by a selector.
GLD for Example 1A

Fig. 6

GLD for Example 1B

Fig. 7
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NV</td>
<td>6</td>
</tr>
<tr>
<td>NROW</td>
<td></td>
</tr>
<tr>
<td>MAXCL</td>
<td></td>
</tr>
<tr>
<td>MAXD</td>
<td></td>
</tr>
<tr>
<td>MAXPAT</td>
<td></td>
</tr>
</tbody>
</table>

**Input for**  
Example IA Section 5

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRITERIAL</td>
<td></td>
</tr>
<tr>
<td>DLIST(NV)</td>
<td></td>
</tr>
<tr>
<td>NE(0:MAXCL)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(NROW,NV)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event array</td>
<td></td>
</tr>
<tr>
<td>Class 0 has 6</td>
<td></td>
</tr>
<tr>
<td>Class 1 has 12</td>
<td></td>
</tr>
<tr>
<td>Class 2 has 11</td>
<td></td>
</tr>
<tr>
<td>Class 3 has 6</td>
<td></td>
</tr>
</tbody>
</table>
GENERALIZE IS OFF
USER Spec IS OFF
PERIM IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE = 100
COMPACT RATIO CRITERIA = 0.25
NUMBER OF VARIABLES = 3
NUMBER OF EVENTS = 15
NUMBER OF CLASSES = 4
MAXIMUM LEVEL = 3
NUMBER OF LEVELS FOR EACH VARIABLE = 4 4 4
NUMBER OF EVENT SPECIFIED FOR EACH CLASS:

CLASS EVENTS
0 6
1 12
2 11
3 6

CLASS FI (1)
EVENT NO. 1* 1 1 0
EVENT NO. 2* 0 2 0
EVENT NO. 3* 0 1 1
EVENT NO. 4* 2 3 0
EVENT NO. 5* 1 2 0
EVENT NO. 6* 0 1 0

CLASS FI (2)
EVENT NO. 1* 3 1 0
EVENT NO. 2* 2 0 0
EVENT NO. 3* 3 2 1
EVENT NO. 4* 2 1 1
EVENT NO. 5* 3 3 2
EVENT NO. 6* 2 2 2
EVENT NO. 7* 1 1 2
EVENT NO. 8* 0 0 2
EVENT NO. 9* 2 3 3
EVENT NO. 10* 1 2 3
EVENT NO. 11* 0 1 3

CLASS FI (3)
EVENT NO. 1* 3 0 0
EVENT NO. 2* 3 1 1
EVENT NO. 3* 2 0 1
EVENT NO. 4* 2 2 1
EVENT NO. 5* 1 1 1
EVENT NO. 6* 0 0 3

CLASS FORMULAS

CLASS 0 GROUP SYM VARS CLASS

FI (1) = (x1*x2’x3’x4) * 0.967 0.500 3 0
FI (1) = (x1’x2’x3’x4) * 1.000 1.000 3 1

THERE ARE NO SYM VARS AND THEREFORE NO SYM FORMULA FOR CLASS 2 0 2
THERE ARE NO SYM VARS AND THEREFORE NO SYM FORMULA FOR CLASS 3 0 3

Output for
Example 1A Section 5
<table>
<thead>
<tr>
<th>Parameter Value</th>
<th>Parameter Type</th>
<th>JCL</th>
<th>Storage Parameters</th>
<th>Optional Parameters</th>
<th>Data Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MANCAT-139</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEN=11B1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>331</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>230</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>310</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>321</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>222</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>332</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>153</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>023</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>310</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>331</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>112</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>233</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>013</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>311</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>201</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>113</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>003</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Input for Example 1B Section 5
Output for
Example 1B Section 5

<table>
<thead>
<tr>
<th>CLASS</th>
<th>EVENTS</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

CLASS F(0)

<table>
<thead>
<tr>
<th>EVENT NO.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>0</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CLASS F(1)

<table>
<thead>
<tr>
<th>EVENT NO.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CLASS F(2)

<table>
<thead>
<tr>
<th>EVENT NO.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CLASS F(3)

<table>
<thead>
<tr>
<th>EVENT NO.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CLASS FORMULI

<table>
<thead>
<tr>
<th>CLASS</th>
<th>COMPACT RATIO</th>
<th>MIN GROUP RATIO</th>
<th>NUM. OF SYM VARS PER CLASS</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(0)</td>
<td>0.567</td>
<td>0.500</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>F(1)</td>
<td>1.000</td>
<td>1.000</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>F(2)</td>
<td>0.917</td>
<td>0.917</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>F(3)</td>
<td>0.600</td>
<td>0.600</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
### Storage Parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERSPEC-3</td>
<td>0.5</td>
</tr>
<tr>
<td>STORAGE</td>
<td>16</td>
</tr>
</tbody>
</table>

### Optional Parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECLIST(1:USERSPEC)</td>
<td>4.3 5</td>
</tr>
</tbody>
</table>

### Data Parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECLIST(1:USERSPEC)</td>
<td>4.3 5</td>
</tr>
</tbody>
</table>

---

#### Example 2A

- Input for Section 5

---

#### Example 2B

- Input for Section 5
Output for Example 2A Section 5

There are 6 symmetric variables and therefore 60 symmetric formula for class 0 0 0.

There are 10 symmetric variables and therefore no symmetric formula for class 1 0 1.
Output for Example 2B Section 5

Class Formulas:

F(0) = (x2+x3+x7^2+3, 5)
F(1) = (x2+x3+x7^2+4)

Class Formulas:

<table>
<thead>
<tr>
<th>Class</th>
<th>MIN COMPACT</th>
<th>O GROUP</th>
<th>SYM VARS</th>
<th>PEP, CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.006</td>
<td>1.000</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.004</td>
<td>1.000</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
```
// EXEC PGM=JENSENZ, REGION=114K
//STEPLIB DD DSN=USER.PC3.JENSENZ,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYIN DD 3

<table>
<thead>
<tr>
<th></th>
<th>JCL</th>
<th>Storage Parameters</th>
<th>Optional Parameters</th>
<th>Data Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DTRED=1/6</td>
<td></td>
<td>0.5</td>
<td>3 4 4 3 5 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 4 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 3 0 1 0 0</td>
<td>1 1 0 4 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 2 0 0 3 1</td>
<td>0 1 0 0 3 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 1 0 1 2</td>
<td>0 2 1 1 2 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 0 0 1 2 2</td>
<td>1 2 0 2 3 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 1 1 1 0 0</td>
<td>1 0 1 2 1 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 3 3 2 4 0</td>
<td>2 2 2 0 4 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1 2 1 3 3</td>
<td>2 3 2 2 2 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 3 1 0 1 2</td>
<td>0 1 3 2 1 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 3 1 0 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Parameter Value</td>
<td>Parameter Type</td>
<td></td>
</tr>
</tbody>
</table>
```

Input for
Example 3 Section 5
### Example 3 Section 5

#### Output for

<table>
<thead>
<tr>
<th>CLASS</th>
<th>MIN</th>
<th>NUM OF SYM WVS</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.040</td>
<td>0.600</td>
<td>5 0</td>
</tr>
<tr>
<td>1</td>
<td>0.067</td>
<td>0.667</td>
<td>5 1</td>
</tr>
<tr>
<td>2</td>
<td>0.035</td>
<td>0.700</td>
<td>5 2</td>
</tr>
</tbody>
</table>

---

**NOTE**: The output table is based on the provided data and represents the classification results for the given data. The columns `CLASS`, `MIN`, and `NUM OF SYM WVS` correspond to the class numbers, minimum values, and number of symmetric weight vectors, respectively.

---

**DATA REDUCTION TABLE**

- **Variable X1** is the sum of variables `X2`, `X3`, `X4`, `X5`, `X6`.
- **Variable X8** is the sum of variables `X1`, `X4`, `X6`.

<table>
<thead>
<tr>
<th>CLASS</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

---

The data reduction table above highlights the relationship between the variables and their corresponding values across different classes. Each row represents a different class, with columns indicating the values for `X1`, `X2`, ..., `X8`. This table is crucial for understanding the behavior and classification outcomes based on the given data set.
6. CONCLUSION

There are different types of symmetry depending upon the value of the reference. The closer the reference is to the minimum or maximum value of the Row_Sum, the less flexibility allowed in the value of the individual variables. At either the minimum or maximum value, the variables are constrained to specific values (i.e., zero or $d_1$ of vars.). If none of the variables are complimented and the reference is in the middle of the Row_Sum range, the symmetric selector can represent the property of 'conservation' between the variables. If one variable is increased, another variable must decrease.

The present application of the SYF4 program is in the area of machine learning. The program is given object descriptions with known class membership, and it generates formulas describing the object classes based upon the symmetries of each class. These formulas represent certain generalizations of the given input information.

Desirable further extensions of this program are

1. Allowing the user to specify more than one group of variables which are to be checked for symmetry. This would eliminate the need to rerun the entire program for multiple groups of user-specified variables.

2. Determining Partial Symmetry in the incompletely specified case. Partial Symmetry is defined as symmetry between some, but not all variables of the $D$-Group. The present version determines Partial Symmetry only in completely specified cases.
3. Allowing the inequality relation in the symmetric selector. The present version allows only the equality relation in the symmetric selector.

4. Allowing the program to generate $\forall L$ formulas containing both symmetric and non-symmetric selectors. The present version generates symmetric selectors only.
REFERENCE


5. Michalski, R.S., "A Geometrical Model for the Synthesis of Interval Covers," Report No. 461, Department of Computer Science, University of Illinois, Urbana, Illinois, June 24, 1971 (in English)

BACKGROUND INFORMATION


APPENDIX A
SOURCE LISTING OF SYSL
PL/I OPTIMIZING COMPILER

MAINP: PROC OPTIONS(MAIN) REORDER;

STMT LEVEL

25 2 0  | (F(3)); ELSE PUT SKIP FILE(SYSPRINT) EDIT
             | (*USERSPEC IS OFF*)(A);
26 2 0  | IF RERUN THEN PUT SKIP FILE(SYSPRINT) EDIT (*RERUN IS ON*)
             | (A); ELSE PUT SKIP FILE(SYSPRINT) EDIT (*RERUN*;
             | * IS OFF*)(A,A);
28 2 0  | IF QTRED THEN PUT SKIP FILE(SYSPRINT) EDIT (*DATA REDUCTION IS ON*)
             | (A,A); ELSE PUT SKIP FILE(SYSPRINT)
             | EDIT (*DATA REDUCTION IS OFF*)(A);
30 2 0  | PUT SKIP FILE(SYSPRINT) EDIT
             | (*MAXIMUM PATTERN SIZE =',MAXPAT,
             | 'COMPACT_RATIO CRITERIA =',CRITERIA1,
             | NOF,'VARIABLES =',NV,
             | NOF,'EVENTS =',NROW,
             | NOF,'CLASSES =',NCL,
             | 'MAXIMUM LEVEL =',MAXL,
             | NOF,'LEVELS FOR EACH VARIABLE =',NCL,
             | NOF,'EVENT SPECIFIED FOR EACH CLASS =',CLASS EVENTS,
             | (ICL,NE(1),ICL) DO ICL=08 TO MAXCL;
             | (*CLASS F(1),ICL 1*),
             | (*EVENT NO. 1,ICL-ROWCL(1),ICL 1*),E1ROW,*
             | DO IROW=ROWCL(1)+18 TO ROWCL(1)+NE(1);
             | DO ICL=08 TO MAXCL;
             | (SKIP,A,F(6)
             | SKIP,A,F(5,2),3 (SKIP,2 A,F(6)),SKIP,A,F(6),SKIP,2 A,
             | (NV)(F(3)),SKIP,F(5),F(10),
             | INCL)(SKIP,A,F(2)),A,NE(1)(SKIP,A,F(5),A,NE(1)(F(5)));
1/* NEW CHECK FOR DISJOINT EVENT CLASSES */
31 2 0  | DO I2=08 TO MAXCL-18;
32 2 1  | ROW1=ROWCL(I2)+18;
33 2 1  | ROW2=ROW1+NE(I2)-18;
34 2 1  | DO I3=ROW1 TO ROW2;
35 2 2  | DO J2=(ROW2+1) TO NROW;
36 2 3  | DO J3=18 TO NV;
37 2 4  | IF E1(I3,J3)=E1(I2,J3) THEN GO TO OK;
38 2 4  | END;
39 2 3  | DO I4=(I2+18) TO MAXCL;
40 2 4  | IF ROWCL(I4) >= J2 THEN GOTO OUT;
41 2 4  | END;
42 2 3  | OUT:
43 2 3  | PUT SKIP FILE(SYSPRINT) EDIT
             | (*THE PROGRAM OPERATES ON DISJOINT EVENT CLASSES*;
             | *EVENT NO. 1,(I3-ROWCL(I2)),* IN CLASS NO. 1*I2,
             | * IS THE SAME AS EVENT NO. 1,(J2-ROWCL(I4))
             | * IN CLASS NO. 1*I4, 'YOUR EVENT CLASSES ARE NOT'*
             | *DISJOINT*; *THIS DATA WILL NOT BE RUN*;
             | (SKIP,A,SKIP,A,F(2),A,F(2)),A,F(2),A,F(2,SKIP,A,A,
             | SKIP,A);
PL/I OPTIMIZING COMPILER

MAIN: PROC OPTIONS(MAIN) READER;

STMT LEVEL

GOTO NEXT;
END;
END;
END;

/* IF THE USER DOES NOT SPECIFY A SPECIFIC SET OF VARIABLES
THE PROGRAM GROUPS ALL THE VARIABLES ACCORDING TO THE
NUMBER OF LEVELS OF THE PARTICULAR VARIABLE. IF THE
USER DOES SPECIFY A SET OF VARIABLES, ONLY THOSE
SPECIFIED WILL BE CONSIDERED FOR SYMMETRY */

IF USERSPEC=OR THEN GOTO LINKER;
NO=3B; DLINK, DTOP, DTOP#=0B;
DO IV=USERSPEC TO LB BY -1B;

DO ID=1B TO NO;

IF DLIST(SPECFLIST(IV))=DLIST(DTOP(ID)) THEN GOTO PUSH01;

END:
ENW01:
NO=ID+1B;

BEGIN:
DTON#(ID)=DTON#(ID)+1B;

DLINK(IV)=DTOP(IV);

DTOP(ID)=SPECFLIST(IV);

END;
ENW01:
DO ID=1A TO NO;

IF DTOP((ID)=1B THEN DO; PUT SKIP FILE(SYSPRINT) EDIT

"YOU HAVE SPECIFIED A SINGLE VARIABLE", IV " AS SYMMETRIC."
" THAT IS ERRONEOUS. CHECK USER "
"SPECFLIST", [A,A,F(A,F),A,A,A];

GOTO TWO;

END;

TWO:
END;

GOTO START;

/* SINCE THERE IS NO USER SPECIFICATION OF VARIABLES, */

GOTO NEXT;

*** PARTITION ALL VARIABLES INTO 'O_GROUPS' WITH SAME #LEVELS */

/* & LINK THE 'O_GROUPS' BY DLINK */

IF ULIST(IV)=DLIST(DTOP(ID)) THEN GOTO PUSH0;

END:
ENW0:
NO=NO+1B;

BEGIN:
DTON#(ID)=DTON#(ID)+1B;

DLINK(IV)=DTOP(IV);

DTOP(ID)=IV;

END;

START:
DLIST=DLIST-LB;
STMT 80 2 0

/* FIND SYMMETRY OF VARIABLES IN EACH CLASS */
DO ICL=0 TO MAXCL;

/* FOR EACH EVENT_CLASS: */
RE_INITIALIZE LISTS,COUNTS &
ICollect 'COLUMN_SUMS' OF EACH VARIABLE.
NOTE -- ALL VARIABLES ARE READ IN 'UNCOMPLETED' FORM,
THEN APPROPRIATE VARIABLES WILL BE COMPLETED
WITH RESPECT TO THEIR MAXIMUM LEVEL -- "D_SLASH"
IT SHOWS THE 'ACTUAL' FORM OF SYMMETRY OF THE VARIABLES.
THEY WILL BE RE_CONVERTED TO 'UNCOMPLETED' FORM
BEFORE THE NEXT EVENT_CLASS IS PROCESSED.

81 2 1 *DSUSTOP,OMAX**=OB; OMAXLINK,OSUBTOP,OSUBTOP**=OB;
83 2 1 ROW1=ROWCL(ICL)+I8;
84 2 1 ARWS=4*ICL;
85 2 1 ROM2=ROW1+RC4S-18;
86 2 1 COMPL=I0*1;
87 2 1 DO IV=1 TO NV;
88 2 2 COLSUM=09;
89 2 2 DO IRCW=ROW1 TO ROW2;
90 2 3 COLSUM=COLSUM+E(IRCIV);
91 2 3 END;
92 2 2 DC51(IV)=COLSUM;
93 2 2 END;

/* FIND SYMMETRY OF VARIABLES IN EACH 'O_GROUP' */
ALL SYMMETRIC VARIABLES HAVE THE SAME NUMBER OF LEVELS WHETHER
COMPLETELY SPECIFIED OR NOT

94 2 1 DO ID=18 TO NO;
95 2 2 /* IGNORE 'O_GROUP' WITH ONLY 1 VARIABLE */
96 2 2 IF DOTP#(ID)=18 THEN GOTO NextO;
97 2 2 N=DOTP#(101); 1
98 2 2 D=DLIST(DOTP#(ID));

/* FIND 'TOTAL SYMMETY' OF ALL VARIABLES IN THE 'O_GROUP'
COMPLEMENT APPROPRIATE VARIABLES.
THE VALUE OF THE SWITCH GEN DETERMINE WHICH COMPLEMENT ROUTINE
IS USED. THE FIRST ROUTINE IS HEURISTIC IN NATURE AND IS DESIGNED
FOR INCOMPLETELY SPECIFIED SYMMETRIES. IT COMPLEMENTS THE VARS
IN DIFFERENT WAYS UNTIL A MINIMUM NUMBER OF ROW SUMS IS PRODUCED.
(IIDEAL ONE ROW SUM)

98 2 2 TOTALSORT:
IF GEN='0'B THEN GOTO OTHER;
PL/I OPTIMIZING COMPILER

STMT LEV NT

99 2 2 IF N>1111D THEN OC; PUT SKIP FILE(SYSPRINT) EDIT
   "THE GENERALIZATION ROUTINE ALLOWS A MAX",
   "UM OF 31 VARIABLES IN ANY D GROUP."
   " THIS JOB IS NOT GENERALIZED.");

101 2 3 GOTO OTHERN;
102 2 3 END;
103 2 2 CPBIN=OB;
104 2 2 MIN=NE(1Cl);
105 2 2 IF NE(1Cl)=18 THEN GOTO OTHERN;

1// GENERATE AND COUNT ROW SUMS
106 2 2 HERE: RS#=0E0B;
107 2 2 DO IRW=ROW1 TO ROW2;
108 2 3 TV=OTOP(IID); IRS=UB;
110 2 3 DO WHILE (IV>0O);
111 2 4 IF COMPL(IV) THEN IRS=IRS+D-E(IV,ROW,IV);
112 2 4 ELSE IRS=IRS+E(IV,ROW,IV);
113 2 4 IV=OLINK(IV);
114 2 4 END;
115 2 3 RS+(IRS)=RS+(IRS)+E0B;
116 2 3 END;
117 2 2 #RS=OB;
118 2 2 DO IRS=OR TO ORH;
119 2 3 IF RS#(IRS)=OB THEN GOTO AGAIN;
120 2 3 #RS=#RS+18;
121 2 3 AGAIN: END;
1// CHECK FOR MINIMUM NUMBER OF ROW SUMS. IF NUMBER OF ROW SUMS
122 2 2 IS EQUALS ONE THEN STOP AND COMPLEMENT EVENTS."
123 2 2 IF #RS=18 THEN GOTO OUT1;
124 2 2 IF MIN=#RS THEN OC;
125 2 2 MIN=RS;
126 2 3 ISAVE=CPBIN;
127 2 2 END;
1// GENERATE A NEW COMPLEMENT VECTOR
128 2 2 CPBIN=CPBIN+18;
129 2 2 IF CPBIN>(2**((N-18)-1H) THEN GOTO OUT2;
130 2 2 IV=OTOP(IID);
131 2 3 DO IV=18 TO N-18;
132 2 3 COMPL(IV)=SUBSTR(UNSPEC(CPBIN),33-IY,18);
133 2 3 IV=OLINK(IV);
134 2 3 END;
1// HAVING GONE THROUGH ALL THE DIFFERENT COMPLEMENT VECTORS, CHOOSE
135 2 2 THE ONE GIVING THE MINIMUM NUMBER OF ROW SUMS
136 2 2 OUT2: CPBIN=ISAVE;
137 2 2 IV=OTOP(IID);
138 2 2 DO IV=18 TO N-18;
139 2 3 COMPL(IV)=SUBSTR(UNSPEC(CPBIN),33-IY,18);
PL/I OPTIMIZING COMPILER

MAINS: PROC OPTIONS(MAIN) REORDER;

STMT LEVEL

139 2 3 | IV=OLINK(IV);
140 2 3 | END;
141 2 2 | /* COMPLEMENT EVENTS USING THE PROPER COMPLEMENT VECTOR */
142 2 2 | IV=OTOP(IV);
143 2 2 | DO WHILE (IV>0);
144 2 3 | IF COMPL(IV)='0' THEN GOTO NEXTV3;
145 2 4 | E(IRW,IV)=0-IRCH,IV);
146 2 4 | END;
147 2 3 | DCS(IV)=D*ROWS-DCS(IV);
148 2 3 | NEXTV3: IV=OLINK(Iv);
149 2 3 | END;
150 2 2 | GOTO TESTPT;


1. LINK VARIABLES WITH SAME 'COLUMN_SUM' IN THE EVENT_CLASS.

2. IF VARIABLE 'IV2' HAS 'COLUMN_SUM' COMPLEMENTARY TO THAT OF 'IV' THEN COMPLEMENT IV2 & LINK IT TO IV'S 'CS_GROUP'. */

151 2 2 | OTHERWISE:
152 2 2 | IV=OTOP(IV);
153 2 3 | DO WHILE (IV>0);
154 2 3 | IF COMPL(IV) THEN GOTO NEXTV;
155 2 3 | IV2=OLINK(IV);
156 2 3 | DO WHILE (IV2>0);
157 2 4 | IF COMPL(IV2) THEN GOTO NEXTV2;
158 2 4 | IF DCS(IV) = D*ROWS-DCS(IV2) THEN GOTO NEXTV2;
159 2 4 | IF DCS(IV) = DCS(IV2) THEN GOTO NEXTV2;
160 2 4 | COMPL(IV2)='1'P;
161 2 5 | DO IRW=ROW1 TO ROW2;
162 2 5 | E(IRW,IV2)=0-IRCH,IV2);
163 2 5 | END;
164 2 4 | DCS(IV2)=DCS(IV);
165 2 4 | NEXTV2: IV2=OLINK(IV2); END;
166 2 3 | NEXTV: IV=OLINK(Iv); END;

/* TEST ENTIRE 'O_GROUP' FOR TOTAL SYMMETRY */
167 2 2 | TESTPT: IF SUF(OTOP(ID),OLINK,N) THEN GOTO VER;
168 2 2 | IF NOTCOMFACTB THEN GOTO SORTCS;
169 2 2 | VER: IF VERIFYD(OTOP(ID),OLINK,N) THEN GOTO TOTALS;
170 2 2 | GOTO SORTCS;

/* ALL VARIABLES IN 'O_GROUP' ARE SYMMETRIC */
171 2 2 | TOTALS: MAXTOP=OTOP(ID);
172 2 2 | MAX*=N;
PL/I OPTIMIZING Compiler  \[\text{MAINP: PROC OPTIONS(MAIN) REORDER;}\]

\[\text{STAT LEV NT}\]

174 2 2 | MAXLINK=DLINK;
175 2 2 | RETLOC=NEXTD; GOTO LINK;

\[\text{/* FURTHER PARTITION VARIABLES OF 'D\_GROUP' INTO 'CS\_GROUPS' */}\]

\[\text{/* WITH SAME COLUMN SUM */}\]

\[\text{/* SORT 'CS\_GROUPS' IN DESCENDING ORDER OF THEIR SIZES,}\]
\[\text{i.e. THE NUMBER OF VARIABLES THEY CONTAIN */}\]

177 2 2 | $\text{SORTCS:} CS=0B; CS\_TOP,CS\_LINK=0B; CS=-18;
180 2 2 | IV=DTCP(1D);
182 2 2 | DO WHILE(IV>0B);
183 2 2 | IF CS(TOP) = DCS(IV) THEN GOTO UPDATE;
184 2 2 | END;
185 2 2 | $\text{NEWCS:} CS=CS+1B;
186 2 2 | CS(TOP)=CS(IV);
187 2 2 | UPDATE: CS#(TOP)=CS#(TOP)+1B;
188 2 2 | CS\_LINK(IV)=CS\_TOP(TOP);
189 2 2 | CS\_TOP(TOP)=IV;
190 2 2 | IV=DLINK(IV);
191 2 2 | END;
192 2 2 | IF CS>18 THEN
193 2 3 | TOP=SORT: DO TOP=1B TO #CS-10;
194 2 2 | IF CS(TOP)>CS(TOP2) THEN GOTO NEXT\_TOP2;
195 2 2 | IBUF=CS(TOP); CS(TOP)=CS(TOP2); CS(TOP2)=IBUF;
196 2 2 | IBUF=CS#(TOP); CS#(TOP)=CS#(TOP2); CS#(TOP2)=IBUF;
197 2 2 | $\text{NEXT\_TOP2: END; END TCP\_SORT;}$

\[\text{/* FIND \text{'TOTAL OR PARTIAL SYMMETRY OF ALL 'CS\_GROUPS'},}\]
\[\text{IGNORE THOSE WITH ONLY 1 VARIABLE}\]
\[\text{& GOTO PROCESS NEXT \text{'D\_GROUP'. */}\]

203 2 2 | TOP=0B;
204 2 2 | NEXTCS: TOP=TOP+1B;
205 2 2 | IF CS(TOP)-1B THEN GOTO NEXTD;
206 2 2 | $\text{IF SUFICS_TOP(TOP),CS\_LINK,CS#(TOP)) THEN GOTO VER2;}$
207 2 2 | IF NOTCOMPACT THEN GOTO PSYM;
208 2 2 | $\text{IF VF\_R\_COMPICS\_TOP(TOP),CS\_LINK,CS#(TOP)) THEN GOTO TSYM;}$
209 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
210 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
211 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
212 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
213 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
214 2 2 | $\text{IF S\_TOP\_2 = S\_TOP\_2 THEN GOTO L, Else}\$
PL/I OPTIMIZING COMPILER

MAINP: PROC OPTIONS (MAIN) REORDER;

STMT COUNT

222 2 2 | AUSD: CO;
223 2 3 | 5*CCLS=5*CCLS-1B;
224 2 3 | IF TCP>K THEN STOP=PT2;
225 2 3 | ELSE $LIST(PT1)=PT2;
226 2 3 | MAX=MAX+1B;
227 2 3 | MAXLINK(K)=MAXTOP;
228 2 3 | MAXTOP=K;
229 2 3 | END;
230 2 2 | K=PT2; IF K>0 THEN GOTO COMPARE;
232 2 2 | IF MAX=1B THEN GOTO TEST;
233 2 2 | IF SUF(MAXTOP, MAXLINK, MAX) THEN GOTO VER3;
234 2 2 | IF NOTCOMPACT THEN GOTO TEST;
235 2 2 | VER3: IF VERIFY(MAXTOP, MAXLINK, MAX) THEN GOTO PSYM1;
236 2 2 | GOTO TEST;

/* SOME BUT NOT ALL VARIABLES IN 'CS_GROUP' ARE SYMMETRIC */
237 2 2 | PSYM1: RETLOC=TEST; GOTO LINK;
239 2 2 | TEST: IF 5*CCLS>10 THEN GOTO NEXT1CS;
240 2 2 | GOTO NEXTICS;

/* ALL VARIABLES IN 'CS_GROUP' ARE SYMMETRIC */
241 2 2 | ITSYM: MAXTOP=CS_TCP(TOP);
242 2 2 | MAX#=CS#(TCP);
243 2 2 | MAXLINK=CS_LINK;
244 2 2 | RETLOC=NEXTICS;
245 2 2 | LINK: OSUPTOP=OSUBTCP+1B;
246 2 2 | PT1=OSUBTCP(0)OSUBTOP*MAXTOP;
247 2 2 | OSUBTOP(0)OSUBTOP#MAX#;
248 2 2 | DO WHILE(PT1>0B);
249 2 3 | PT2=PT1;
250 2 3 | PT1=MAXLINK(PT1);
251 2 3 | OMAXLINK(PT2)=PT1;
252 2 3 | END;
253 2 2 | OMAX#=OMAX#MAX#;
254 2 2 | GOTO RETLOC;

/* VERIFY: VERIFIES THAT OTHER CLASSES */
/* DO NOT HAVE THE SAME SYMMETRY AS THE CURRENT CLASS. */
/* I.E. THE CURRENT SYMMETRIC SELECTOR IS ACCEPTED */
/* IF IT DOES NOT APPLY TO OTHER CLASSES. */
255 2 2 | VVERIFY: PROC(NODELPT, LIST, *CCLS) RETURNS(BIT(1));
256 3 2 | OQL(NODELPT, LIST(*), *CCLS, ICL2, KHS, KROW) FIXED BIN;
257 3 2 | DO ICL2=0B TO ICL-1R, ICL+1B TO MAXCL;
258 3 3 | DO IROW=1B TO NET(1CL2);
```plaintext
OPTIMIZING COMPILER

MAINP: PROC OPTIONS(PMAIN) REORDER;

STMT LEVEL

259 3 4 | KRS=OB; KRCW=IROW+ROWCL(1CL2);
261 3 4 | IV=MODELPt;
262 3 4 | DO WHILE(IV<OB);
263 3 5 | IF COMPL(IV) THEN KRS=KRS+D-E(KROW,IV);
264 3 5 | ELSE KRS=KRS+ E(KROW,IV);
265 3 5 | IV=LIST(IV);
266 3 5 | END;
267 3 4 | IF INCOPL(KRS) THEN RETURN('O'B);
268 3 4 | IF KRS(KRS)<>OB THEN RETURN('O'B);
269 3 4 | END;
270 3 3 | END;
271 3 2 | INCOPL='O'B; RETURN('I'B);
273 3 2 | END VERIFY;

/* 'SUFZ' FINDS PAIRWISE SYMMETRY OF 2 VARIABLES J & K */

274 2 2 | SUFZ: PROC(J,K) RETURNS(BIT(1));
275 3 2 | RS=OE04;
276 3 2 | DO IROW=ROW1 TO ROW2;
277 3 3 | IRS=4(IRCW,J)+E(IRCW,K);
278 3 3 | RS*(IRS)=RS*(IRS)+EOB;
279 3 3 | END;

/* CHECK ALL 'ROW_SUMS' FOR SUFFICIENT OCCURRENCES */

280 3 2 | DO IRS=OB TO D*100;
281 3 3 | IF RS*(IRS) = OEOO THEN GOTO NEXT_RS;
282 3 3 | #COMP=COMP(10B,IRS,0);
283 3 3 | IF #KRCW/#COMP < CRITERIAL /* NO NEED TO CHECK THIS GROUP OF VARIABLES SINCE COMPACT_RATIO CRITERIAL IS NOT SATISFIED */
284 3 3 | IF #COMP>MAXPAT
285 3 4 | THEN /* PATTERN SIZE EXCEEDS MAXPAT */ DO:
286 3 4 | PUT FILE(SYSPRINT) EDIT
287 3 4 | ('PATTERN SIZE=',#COMP,' EXCEEDS MAXPAT=',MAXPAT)
288 3 4 | ('SKIP2 A-111));
289 3 4 | RETURN('O'B);
290 3 4 | END;
291 3 3 | BEGIN;
292 4 3 | OCL (LISTS(#COMP,108),LEVEL(108),ICOMP,ICOL)
293 4 3 | FIXED BIN;
294 4 3 | /* CALL 'RSLIST' TO GENERATE ALL POSSIBLE PATTERNS. */
295 4 3 | CALL RSLIST(108,IRS,LISTS,LEVEL);
296 4 3 | DO #COMP=18 TO #COMP;
297 4 4 | DO IROW=RCW1 TO RCW2;
298 4 4 | DU ICOL=1A TO 10B;
```
PL/I OPTIMIZING COMPILER

STMT: LEV NT

294 4 6 IF ICCL=1B THEN IV=J; ELSE IV=K;
295 4 6 IF EI(IV,IV)=LISTA(ICOMP,ICOL)
296 4 6 THEN GOTO NEXT_ROW;
297 4 6 END;
298 4 5 /* MATCH THIS PATTERN */ GOTO NEXT_PATTERN;
299 4 5 INEXT_ROW: END;
300 4 4 /* NOT MATCH THIS PATTERN */
301 4 4 INEXT_PATTERN: END;
302 4 3 /* MATCH ALL PATTERNS */
303 3 3 INEXT_RS: END;
304 3 2 RETURN('1'B);
305 3 2 ENC SUF2:

1/* 'SUFR TESTS 'TOTAL SYMMETRY' OF A LIST OF VARIABLES
1. #OCURRENCES OF EXISTING 'ROW_SUMS' ARE COLLECTED
2. THE VARIABLES ARE 'TCTALLY SYMMETRIC'
   IF THE EXISTING 'ROW_SUMS' ARE COMPLETE--
   I.E. THEY OCCUR 'SUFFICIENTLY' IN ALL POSSIBLE WAYS.
ELSE 3. THE INCOMPLETE 'ROW_SUMS' ARE CHECKED FOR CLOSENESS TO
   COMPLETENESS UNDER A 'COMPACT_RATIC' CRITERIA SUPPLIED BY USER
   IFSO THE SYMMETRIC SELECTOR WITH THESE 'ROW_SUMS'
   WILL BE 'COMPACT' AND THEREFORE ACCEPTABLE
   NOTE THIS SYMMETRIC SELECTOR STILL HAS TO BE 'VERIFIED'
   THAT IT DOES NOT APPLY TO OTHER EVENT_CLASSES.
306 2 2 SUFR: PROC (NODEP,LIST1,#CCLS) RETURNS(BIT11);
307 3 2 DCL (NODEP,LIST1,#CCLS) FIXED BIN,
308 3 2 SUFR BIT BIT11 ALIGNED INIT('1'B);
309 3 2 NOTCOMPACTB='0'B; INCLUDE='0'B; RS#='0'OB;
310 3 2 DO IRW=ROW1 TO ROW2;
311 3 2 DO IV=NODEP TO IRS='OB;
312 3 3 DO WHILE(IV<OB);
313 3 4 IRS=IRS+1(IV,IV);
314 3 4 IV=LIST11(IV);
315 3 4 END;
316 3 4 RS#(IRS)=RS#(IRS)+1OB;
317 3 3 END:
318 3 3 END;
319 3 3 /* CHECK ALL 'ROW_SUMS' FOR SUFFICIENT OCCURRENCES */
320 3 2 SUFRS: DO IRS=OR TO D#CCLS;
321 3 3 IF IRS#(IRS) = '0'OB THEN GOTO NEXT_RS;
322 3 3 WCOMP=COMP('COLS,IRS#); 323 3 3 IF #ROWS/#CCMP < CRITERIAL 324 THEN/
   /* NO NEED TO CHECK THIS GROUP OF VARIABLES

52
Since COMPACT_RATE CRITICAL is not satisfied

DO:
   NOTCOMPACT3='1'b; RETURN('0'b);
END;

IF NOTCOMPACT3='1'b;
THEN /* PATTERN SIZE EXCEEDS MAXPAT */
   DO:
      NOTCOMPACTB='1'b;
      PUT FILE(SYSPRINT) EDIT
         ('PATTERN SIZE=',#COMP,' EXCEEDS MAXPAT=',MAXPAT)
         (SKIP,2 (A,FILL));
      RETURN('0'b);
   END;
BEGIN;
BEGIN:
   DCL (LISTS[#COMP,#COLS],LEVEL[#COLS],ICOMP,ICOL,NTOCCUR);
   FIXED BIN;
   NTOCCUR=08;
   /* CALL 'RSLIST' TO GENERATE ALL POSSIBLE PATTERNS. */
   CALL 'RSLIST'(#COLS,#LISTS,#LEVEL);
   DO ICOMP=18 TO #COMP; /* CHECK ALL PATTERNS */
   DO IROW=ROW1 TO ROW2; /* AGAINST ALL EVENTS IN CLASS. */
   IF #E(IROW,IV)<=LISTS(ICOMP,ICOL) THEN GOTO NEXT_ROW;
   END;
   IV=NCDELT;  
   ON ICOL+18 TO #COLS;
   IF #E(IROW,IV)<=LISTS(ICOMP,ICOL) THEN GOTO NEXT_ROW;
   END;
   IV=NCDELT;  
   ON ICOL+18 TO #COLS;
   IF #E(IROW,IV)<=LISTS(ICOMP,ICOL) THEN GOTO NEXT_ROW;
   END;
   /* MATCH THIS PATTERN */
   GOTO NEXT_PATTERN;
   END;
   /* MATCH THIS PATTERN */
   SUBFIB='0'b;
   INCOMP=1cRIS='1'b;
   NOTOCUR#NOTOCUR#='1'b;
   RATIO=1#CCMP-NOTOCUR#/NOTOCUR;
   IF RATIO>CRITICAL THEN GOTO NEXT_PATTERN;
   NOTOCMPACTB='1'b; RETURN('0'b);
END;
/* CALL 'RSLIST' TO GENERATE ALL POSSIBLE PATTERNS */
OPTIMIZING COMPILER

STMT LEVEL

1/* FOR N VARIABLES, WITH SAME MAXIMUM LEVEL -- 0, 
ITU SUM TO IRS -- 'ROW_SUM' & PUT THEM IN LISTS(*, *) . */

357 2 2 IRSLIST: PROC(N,IRS,0,LISTS,LEVEL):
358 3 2 | DCL (LISTS(*,*),LEVEL(*)) FIXED BIN:
359 3 2 | DCL (IRG,N,IRS,0,NV,R,IV) FIXED BIN:
360 3 2 | IRW,NV=1B:
361 3 2 | NEXTV: IF N=NV THEN DO; R=IRS:
362 3 3 | DO IV=1B TO N-1B:
363 3 4 | R=R-LEVEL(IV);
364 3 4 | END:
365 3 4 | IF R>Q THEN GOTO POP;
366 3 3 | LEVEL(N)=R;
367 3 3 | LISTS(IRW,*)=LEVEL;
368 3 3 | IRW=IRW+1B:
369 3 3 | GOTO POP;
370 3 3 | END;
371 3 3 |
372 3 2 | LEVEL(NV)=Q0;
373 3 2 | PUSH: NV=NV+1B:
374 3 2 | GOTO NEXTV:
375 3 2 | POP: NV=NV-1B:
376 3 2 | IF NV=Q0 THEN RETURN;
377 3 2 | IF LEVEL(NV)=0 THEN GOTO POP;
378 3 2 | LEVEL(NV)=LEVEL(NV)+1B:
379 3 2 | GOTO PUSH;
380 3 2 | END RSLIST;

1/* 'COMP' CALCULATES THE NUMBER OF WAYS FOR N VARIABLES 
1/* WITH SAME MAXIMUM LEVEL--0, TO SUM TO R--'ROW_SUM' */

381 2 2 COMPC: PROC(N,R,0) RETURNS(FLOAT BIN(53));
382 3 2 | DCL (N,R,0,M,1,MAX,RL,*2) FIXED BIN(15),
383 3 2 | (SUM,SIGN) FLOAT BIN(53); 
384 3 2 | SIGN=-1E0B; SUM=Q0B;
385 3 2 | RI=R; R2=0QN-R1;
386 3 2 | IF RL>R2 THEN RL=R2;
387 3 2 | MAX=RL/(Q+1B);
388 3 2 | DO RL=Q0 TO MAX:
389 3 2 | I=RL-Q(*Q+1B);
390 3 3 | SIGN=-SIGN;
391 3 3 | SUM=SUM+SIGN*COMB(N,M)*COMB(N-I-1B,1);
392 3 3 | END;
393 3 3 |
394 3 2 | RETURN(SUM);

1/* 'COMB' CALCULATES THE NUMBER OF WAYS TO CHOOSE M OUT OF N OBJECTS*/

395 3 2 COMB: PROC(N,M) RETURNS (FLOAT BIN(53));
396 4 2 | DCL (N,M,TIMES,NL,N2) FIXED BIN(15),
397 4 2 | (TOTAL,FACM) FLOAT BIN(53);
IF \( M=0 \) THEN RETURN(IEOB);  
IF \( M=* \) THEN RETURN(N);  
IF \( M=*'0B \) THEN RETURN((N\*(N-1))\!/10B );  
TOTAL,FAC\*=10B;  
N1=N+1B-M; N2=N-M;  
IF \( N-M \geq 4 \) THEN GOTO CALC;  
NI=N+1B; N2=N-M;  
IF N1 \( \geq T \) THEN GOTO CALC;  
TOTAL*FAC\*=TOTAL*TIMES;  
END:  
N1=N1-N2;  
IF N1 \( \geq T \) THEN GOTO CALC;  
TOTAL=TOTAL*TIMES;  
END;  
FAC=TOTAL/FAC\*-TIMES;  
END;  
RETURN(TOTAL/FAC\*-TIMES);  
END COMB;
SYMVAR('IV')='1'B;

COLS='LB'; IF IV>10018 THEN COLS='10B';

IF COMPL('IV') THEN DO; PUT FILE(SYSPRINT) EDIT
  (CHR(SELECTR),*X',IV,')
  (A,A,F(COLS),A);
  COUNT=COUNT+3+COLS; END;

ELSE DO; PUT FILE(SYSPRINT) EDIT
  (CHR(SELECTR),*X',IV)
  (A,A,F(COLS));
  COUNT=COUNT+2+COLS; END;

IF SElCHR='1B THEN SElCHR='10B';

IF COUNT>70 THEN DO; PUT SKIP FILE(SYSPRINT) EDIT
  (IV=DMAXLINK('IV');
  IV=DMAXLINK('IV');

END;

END;

RS*='0E0B;

SElCHR='11B';

DO IRC='ROW1 TO ROW2';

IV=DSUBTOPIOP); IRS='0B';

DO WHILE(IV>0B);

IRS=IRS+6(IROW,IV);

IV=DMAXLINK('IV');

END;

RS*(IRS)='K*(IRS)*L0B';

END;

N=D0SUBTOP+1OP); D=DLIST(DSUBTOP+1OP);

DO IRS=08 TO 0*N;

IF RS*(IRS)='0E0B THEN G0TO NEXTRS;

COLS='LB'; IF IRS>10018 THEN COLS='10B';

PUT FILE(SYSPRINT) EDIT
  (CHR(SELECTR),IRS+1A,F(COLS));

COUNT=COUNT+1B+CCLS;

IF SElCHR='11B THEN SElCHR='10B';

/* CALCULATE CLASS COMPACT RATIO AND THE DGROUP (OR SELECTOR) COMPACT RATIO */

#COMP=#COMP(N,IRS,0);

#COMB=#COMB*#COMP;

#REPETITIONS=RS*(IRS)/#CMIP;

IF MC0(RS*(IRS),#CMIP) => 0 THEN

#REPETITIONS=#REPETITIONS+1B;

RATIO=RS*(IRS)/#REPETITIONS*#CMIP;

IF RATIO<MINRATIO THEN MINRATIO=RATIO;

NEXTRS;

END;

#COMB=#COMB*#COMP;

PUT FILE(SYSPRINT) EDIT('
  (IV=DMAXLINK('IV);

COUNT=COUNT+1B;

END;

DO IV=1B TO NV;
IF \( \text{SYMVAR}(	ext{IV}) \) THEN

\[
\text{DCOMB} \times \text{CCOMB} \times (\text{DLIST}(\text{IV}) \times \text{EOO}) ;
\]

/* PRINT KATICS */

PUT FILE(SYSPRINT) EDIT

\[
\{ \text{ROWS} / \# \text{CCOMM}, \# \text{MMRATIO}, \# \text{OMAX}, \# \text{ICL} \} \times (85-\text{COUNT}), \text{F}(4,1),
\]

\[
\text{X}(4), \text{F}(6,3), \text{X}(7), \text{F}(3,1), \text{X}(6), \text{F}(2) ;
\]

/* RE_CONVERT THE VARIABLES IN THIS EVENT_CLASS */

/* NOTE — THIS IS REQUIRED WHEN FORMULA FOR LATER CLASSES ARE VERIFIED AGAINST THIS EVENT_CLASS */

UNPRIME: DO IO=1A TO NO;

IV2, IV=DTOP(10);

DO WHILE (IV>08);

IF COMPL(IV) THEN

DO IRW=ROW1 TO ROW2;

E(IRW, IV) = OLIST(IV2) - E(IRW, IV);

END;

END;

UNCOMPLE: DO IO=1A TO NO;

IV2, IV=DTOP(10);

DO WHILE (IV>08);

IF COMPL(IV) THEN

DO IRW=ROW1 TO ROW2;

E(IRW, IV) = OLIST(IV2) - E(IRW, IV);

END;

END;

END;

IF OMANT=08 THEN RERUN=*18;

/* GENERATE LIST OF SYMMETRIC VARIABLES CCMMCN TO ALL CLASSES WHICH */

IS USED IN THE DATA REDUCTION TABLE */

IF \( \text{OMANT} \) THEN GOTO NEWCLASS;

IF ICL=08 THEN GOTO CHECK;

SCOMPL=COMPL; SDOMAXLINK=DOMAXLINK;

SDSUBTP=DSUBTP; SODSUBTP=DSUBTP;

SDSUBTOP=DSUBTOP; SDSUBTOP=SODSUBTOP;

GOTO NEWCLASS;

IF SORCC*08 THEN GOTO NEWCLASS;

DO I=1B TO SDSUBTOP;

DO J=1B TO SODSUBTOP;

IF DDSUBTOP*08 THEN GOTO A3;

END;

GOTO A4;

IF DDSUBTOP*08 THEN GOTO A3;

END;

GOTO A4;
58

PL/I OPTIMIZING COMPILER

MAIN: PROC OPTIONS(MAIN) RECODE;

STMT LEVEL

528 2 1 NEWCLASS: END;

529 2 0 IF RERUN THEN DO; DLIST=DLIST+1;
530 2 1 PUT SKIP FILE(SYSPRINT) EDIT('THIS JOBS IS RERUN',
531 2 1 'WITH THE SAME EVENTS AND AUTO SELECTION',
532 2 1 'OF D GROUPS')(A;A;A);
533 2 1 GOTO LINKER;

534 2 0 IF REDUCTION THEN GOTO NEWDATA;
535 2 0 IF SDSUBTOP=08 THEN DO; PUT SKIP FILE(SYSPRINT) EDIT
536 2 0 ('THERE ARE NO SETS OF SYMMETRIC VARIABLES',
537 2 0 'COMMON TO ALL CLASSES. THERE IS',
538 2 0 'NO DATA REDUCTION AND THEREFORE',
539 2 0 'NO DATA REDUCTION TABLE')(A;A;A);
540 2 1 GOTO NEWDATA;
541 2 1 END;

542 2 1 DO 1=08 TO NV;
544 2 1 IF SDSUBTOP(I)=08 THEN GOTO AIS;
543 2 1 J=08;
545 2 1 IF (NV+J)<10108 THEN CL=18; ELSE CL=108;
546 2 1 PUT SKIP FILE(SYSPRINT) EDIT ('VARIABLE X',NV+J,' IS THE SUM',
547 2 1 'OF VARIABLES')(A;F(CL);A;
548 2 1 IVL=SDSUBTOP(I); SLC=18;
549 2 1 DO WHILE (IVL<08);
550 2 2 DV=DLIST(IVL);
551 2 2 IF IVL<10108 THEN DL=18; ELSE DL=108;
552 2 2 IF SCOMP(VL) THEN DO; PUT FILE(SYSPRINT) EDIT
553 2 2 (CHAR2(SLC),'X',IVL,'**')((A;A;F(DL);A);
554 2 3 EV(*,J)=EV(*,J)+DV-E(*,IVL));
555 2 3 END;
556 2 3 ELSE DO; PUT FILE(SYSPRINT) EDIT
557 2 2 (CHAR2(SLC),'X',IVL)((A;A;F(DL));
558 2 3 EV(*,J)=EV(*,J)+E(*,IVL);
IF SLC=18 THEN SLC=10B;
N1=5J\text{MAXLINK}(N1);
END;

IF I>10018 THEN CL=10B; ELSE CL=1B;

PUT SYSFILE(SYSPRINT) EDIT (*',)((SKIP,SKIP,X(14)),A);

DO I=1B TO NV;

IF I>1001B THEN CL=10B; ELSE CL=1B;

PUT SYSFILE(SYSPRINT) EDIT (*',)((X(100B-CL)),A,CL);

END;

PUT SYSFILE(SYSPRINT) EDIT (*',)(A);

DO I=NV+1B TO NV+J;

IF I>1001B THEN CL=10B; ELSE CL=1B;

PUT SYSFILE(SYSPRINT) EDIT (*',),(X(100B-CL)),A,CL);

END;

PUT SYSFILE(SYSPRINT) EDIT

("CLASS F',A,F(2),A,(NE(4)),A,SKIP,A,F(5)),A,(NV)(F(5)),A,
(J)(F(5)))

NEWDATA: END;

GOTO NEXT;

EXIT: END MAINP;