75-7

75-7

UIUCDCS-R-75-774

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 SYM4 program and its earlier version. In particular, Professor R. S. Michalski for his helpful suggestions and guidance throughout the development of the SYM4 program, Thomas Tan 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 74-03514.

TABLE OF CONTENTS

1. INTRODUCTION

This paper contains a description of an algorithm that detects symmetry among variables of a VL_1 function, and a users guide for the program SYM4 which is the PL/1 implimentation 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 paper[1]. 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 $d_i - \dot{x}_i$, where d_i is the maximum domain element for the i^{th} domain and $\dot{x}_i$ 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 <u>symmetric</u> <u>selector</u> is a selector in which there is more than one literal in the referee (L). The following is an example of a symmetric selector:

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

<u>Domains of Input Variables</u>

$$D_i = \{0,1,2,3,4\}$$
$$i=1,2,5$$

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 ith component of events in one event class. The compliment of a Column_Sum is defined as the sum of the compliments of ith 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., # events * $\not{d}_i$ - Col-Sum$_i$=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 complimented, its value is the complimentary value of the variable $\not{d}_i - x_i$.

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 N variables with the same maximum value ($\not{c}$) can sum to R (Row_Sum_i) the following is computed. First choose the minimum of the Row_Sum_i or Maximum $Row\_Sum - Row\_Sum_i$, 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., $M = \left\lfloor \frac{Row\_Sum^*}{\#\ of\ Levels} \right\rfloor$). The number of ways N variables sum to R is then

$$\sum_{i=0} (-1)^i \binom{N}{i} \binom{N+(X_i-1)}{X_i}$$

where X_i is initially the value of Row_Sum^* and is decreased by the # of levels for each term in the sum ($X_i = Row\_Sum^* - i*(\#\ of\ Lev)$).

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 <u>no</u> 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 pariticular 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_Groups 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.

<u>End of Algorithm</u>

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.

<u>AND</u>

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

<u>OR</u>

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.

<u>OR</u>

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

<u>AND</u>

Events in all other event classes do not satisfy the condition $x_1+x_2+x_3=3$.

3. USER'S GUIDE FOR SYM4

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 PL/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 PL/1 data format) which specify the options that should be used during program execution. Since these parameters are specified in PL/1 data foramt, 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 PL/1 list format) which specify the VL_1 functions which are checked for symmetry. Since these parameters are also in PL/1 list format, they have no default values and must be presented in the order specified.

3.2 Specification of Storage Parameters

- NV

Example: $\boxed{7}$

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: $\boxed{16}$

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: $\boxed{1}$

Possible Values: Any positive integer $\leq$ 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: $\boxed{3}$

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 occurances 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

- MAXPAT

Example: MAXPAT=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).

- GEN

Example: GEN='0'B

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

If GEN='1'B, a heuristic routine is used to decide which variables should be complimented. The heuristic used is as follows: the variables in the D_Group should be complimented to achieve a minimum number of different Row_Sums (ideally one).

If GEN='0'B, a variable is complimented 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.

- USERSPEC

Example: USERSPEC=0

Possible Values: Any positive integer $\leq$ NV

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.

- RERUN

Example: RERUN='0'B

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

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 $\leq$ MAXD + 1

This parameter specifies the number of levels (or possible values) DLIST(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).

● NE (0:MAXCL)

Example: $\boxed{10,6}$

Possible Values: A list of MAXCL + 1 positive integers which sum to the value of 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.

● E (NROW,NV)

Example:

```
1 0 1 3 1 3 0
0 0 2 1 3 3 1
3 1 0 3 [illegible] 2 0
1 1 1 1 2 3 1
1 1 2 0 1 0 2
2 2 0 1 0 0 1
2 2 1 2 0 1 2
2 1 2 2 0 3 0
3 2 1 1 0 2 0
3 2 2 0 0 0 1

0 0 2 3 2 2 0
0 1 1 1 2 2 0
1 1 2 3 3 2 1
1 2 0 0 0 3 0
2 2 1 3 0 0 1
2 2 2 0 2 2 2
```

Possible Values: A set of non-negative integers which represent NROW events of NV variables.

E is a list of events defining the given VL_1 function. There are no default values and each value corresponding to the value of variable x_i must be less than DLIST(i).

The following parameter is specified if USERSPEC≠0. It is specified in PL/1 list format.

● SPECILIST (USERSPEC)

Example: $\boxed{1,\ 3,\ 4}$

Possible Values: A list of USERSPEC, positive integers with unique values = NV.

This parameter is specified if USERSPEC $\neq$ 0. USERSPEC specifies how many variables should be checked for symmetry. SPECLIST is the list of variable subscript numbers used to specify which variables should be checked. No duplication of variables is allowed in SPECLIST.

This is the end of the input parameters for one problem. Several problems may be run at one time by specifying all the parameters in succession. The default value of the Optional Parameters are the only parameters carried over from one problem to another.

3.5 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.

● FORMULAS

Example:

```
          CLASS FORMULA

F( 0) := (X2+X3+X7=3,5) & (X1+X4+X5+X6=10)
F( 1) := (X2+X3+X7=4) & (X1+X4+X5+X6=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:

*****DATA REDUCTION TABLE*****

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

		X1	X2	X3	X4	X5	X6	X7	X8	X9
CLASS F(0)										
EVENT NO.	1=	1	0	1	3	3	3	2	3	10
EVENT NO.	2=	3	0	2	1	3	3	1	3	10
EVENT NO.	3=	3	1	0	3	1	3	2	3	10
EVENT NO.	4=	3	1	1	3	3	1	1	3	10
EVENT NO.	5=	2	1	2	2	3	3	0	3	10
EVENT NO.	6=	2	2	0	3	2	3	1	3	10
EVENT NO.	7=	2	2	1	3	3	2	0	3	10
EVENT NO.	8=	3	1	2	2	2	3	2	5	10
EVENT NO.	9=	3	2	1	2	3	2	2	5	10
EVENT NO.	10=	3	2	2	3	2	2	1	5	10
CLASS F(1)										
EVENT NO.	1=	2	0	2	3	3	3	2	4	11
EVENT NO.	2=	3	1	1	2	3	3	2	4	11
EVENT NO.	3=	3	1	2	3	2	3	1	4	11
EVENT NO.	4=	3	2	0	3	3	2	2	4	11
EVENT NO.	5=	3	2	1	3	3	3	1	4	12
EVENT NO.	6=	0	2	2	0	0	0	0	4	0

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 paper[2].

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 reference[3], and using the above assumptions on the evaluation of formulas. The instruction times used are as follows:

Add time = 0.68 μsec.
Subtract time = 0.68 "
Compare time = 0.68 "
Branch time = 1.01 "

Time Required to Evaluate Formulas
(All times in microseconds)

		Class 0	Class 1	Class 2	Class 3
Example 1	AQVAL	13.52	28.73	27.04	15.21
	SYM4	5.42	3.73	3.73	3.73

		Class 0	Class 1
Example 2	AQVAL	18.59	27.04
	SYM4	8.47	10.76

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

The AQVAL program stores complexes 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 MOD_8(Total # Levels) bytes. The SYM4 program stores its formulas as a list of variable subscripts, a list of references and a compliment 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.

Storage Required
(in bytes)

		Class 0	Class 1	Class 2	Class 3	Total
Example 1	AQVAL	6	20	16	12	54
	SYM4	8	7	7	7	29

		Class 0	Class 1	Total
Example 2	AQVAL	16	20	36
	SYM4	17	16	35

In most cases, the SYM4 formulas required less storage.

*For further information--see paper[2]

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 (arithmatic sum) between particular variables. If the given input information conains an incompletely specified symmetry, the symmetric selector covering it represents a generalization of that information. This generalization is considered machine learing in paper[4]. 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
RERUN IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE    =   100
COMPACT_RATIO CRITERIA = 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         6
    1        12
    2        11
    3         6
CLASS F( 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 F( 1)
EVENT NO.     1=    3    2    0
EVENT NO.     2=    2    1    0
EVENT NO.     3=    1    0    0
EVENT NO.     4=    3    3    1
EVENT NO.     5=    2    2    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 F( 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 F( 3)
EVENT NO.     1=    3    0    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
```

SYM4 Output for
Example 1 Section 4

Fig. 1

CLASS FORMULI	CLASS COMPACT RATIO	MIN D GROUP RATIO	NUM. OF SYM VARS PER CLASS	CLASS
F(0) := (X1+X2'+X3=1,2)	0.667	0.500	3	0
F(1) := (X1+X2'+X3=4)	1.000	1.000	3	1
F(2) := (X1+X2'+X3=5)	0.917	0.917	3	2
F(3) := (X1+X2'+X3=6)	0.600	0.600	3	3

```
DISJOINT COVERS      FACTOR VARS

SPACE ALLOCATED FOR GIFI IS  200                SPACE ALLOCATED FOR MQ IS   75

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   SAVELQ=  0

INPUT FORMAT IS VECTOR

ALL VARIABLES WILL BE COVERED BY FACTORS

NUMBER OF VARIABLES =   3

NUMBER OF LEVELS FOR EACH VARIABLE:  4  4  4

NUMBER OF EVENTS SPECIFIED FOR EACH CLASS:
        CLASS   #EVENTS
          0        5
          1       12
          2       11
          3        6

                              **  AMOUNT OF UNUSED CORE =   0K  **


OLIST=    0   1   2   3

NCRIT=    2
CLIST=    1        2
TLIST=    0.00     0.00

LQ-STAR OPTION  LOST=   FIFO

CLASS FI 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 FI 1)
     EVENT NO.   1=     3     2     0
     EVENT NO.   2=     2     1     0
     EVENT NO.   3=     1     0     0
     EVENT NO.   4=     3     3     1
     EVENT NO.   5=     2     2     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 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     3
     EVENT NO.   5=     1     1     3
     EVENT NO.   6=     0     0     3
```

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

Fig. 2

Note: Figure 2 is continued on the following two pages.

THE FOLLOWING [illegible] CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 0

	COV	NEW	IND	TOT
COMPLEX: (X1= 0 1) (X2= [illegible]) (X3= 0 1)	4	4	2	4
COMPLEX: (X2= 3) (X3= 0)	2	1	1	5
COMPLEX: (X1= 3) (X3= 0)	2	1	1	6

**** DELTA FOR THIS SET IS 2 ****

THE LARGEST STAR HAD 5 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 9 ELEMENTS

THE FOLLOWING 10 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 1

	COV	NEW	IND	TOT
COMPLEX: (X1= 3) (X2= [illegible]) (X3= 0)	1	1	1	1
COMPLEX: (X1= 2) (X2= 1) (X3= 0)	1	1	1	2
COMPLEX: (X1= 1) (X2= 0 1) (X3= 0 1)	2	2	2	4
COMPLEX: (X1= 3) (X2= 3) (X3= 1)	1	1	1	5
COMPLEX: (X1= 2) (X2= 2) (X3= 1)	1	1	1	6
COMPLEX: (X1= 0) (X2= 0) (X3= 1)	1	1	1	7
COMPLEX: (X1= 0 2) (X2= 1 3) (X3= 2)	2	2	2	9
COMPLEX: (X1= 1) (X2= 2) (X3= 2)	1	1	1	10
COMPLEX: (X1= 1) (X2= 3) (X3= 3)	1	1	1	11
COMPLEX: (X1= 0) (X2= 2) (X3= 3)	1	1	1	12

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

THE LARGEST STAR HAD 5 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 9 ELEMENTS

THE FOLLOWING [illegible] CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 2

	COV	NEW	IND	TOT
COMPLEX: (X1= 3) (X2= 1) (X3= 0)	1	1	1	1
COMPLEX: (X1= 2) (X2= 0 2) (X3= 0 2)	2	2	2	3
COMPLEX: (X1= 3) (X2= 2) (X3= 1)	1	1	1	4
COMPLEX: (X1= 2) (X2= 1 3) (X3= 1 3)	2	2	2	6
COMPLEX: (X1= 1 3) (X2= 1 3) (X3= 2)	2	2	2	8
COMPLEX: (X2= 0) (X3= 2)	1	1	1	9
COMPLEX: (X1= 1) (X2= 2) (X3= 3)	1	1	1	10
COMPLEX: (X1= 0) (X2= 1) (X3= 3)	1	1	1	11

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

Figure 2 (cont'd)

THE LARGEST STAR HAD 2 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 5 ELEMENTS

THE FOLLOWING 6 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS 3

				COV	NEW	IND	TOT
COMPLEX:	(X1= 3)	(X2= 0)	(X3= 0)	1	1	1	1
COMPLEX:	(X1= 3)	(X2= 1)	(X3= 1)	1	1	1	2
COMPLEX:	(X1= 2)	(X2= 0)	(X3= 1)	1	1	1	3
COMPLEX:	(X1= 2)	(X2= 2)	(X3= 3)	1	1	1	4
COMPLEX:	(X1= 1)	(X2= 1)	(X3= 3)	1	1	1	5
COMPLEX:	(X2= 0)	(X3= 3)		1	1	1	6

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

THE LARGEST STAR HAD 2 ELEMENTS; THE LARGEST INTERMEDIATE STAR HAD 5 ELEMENTS

*** NORMAL TERMINATION ***

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

Fig. 2 (continued)

```
GENERALIZE IS OFF
USERSPEC IS OFF
RERUN IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE    =   300
COMPACT_RATIO CRITERIA  = 0.50
NUMBER OF VARIABLES =      7
NUMBER OF EVENTS    =     16
NUMBER OF CLASSES   =      2
MAXIMUM LEVEL       =      3
NUMBER OF LEVELS FOR EACH VARIABLE :  4   3   3   4   4   4   3
NUMBER OF EVENT SPECIFIED FOR EACH CLASS :
CLASS    #EVENTS
    0        10
    1         6
CLASS F( 0)
EVENT NO.    1=    1    0    1    3    3    3    2
EVENT NO.    2=    3    0    2    1    3    3    1
EVENT NO.    3=    3    1    0    3    1    3    2
EVENT NO.    4=    3    1    1    3    3    1    1
EVENT NO.    5=    2    1    2    2    3    3    0
EVENT NO.    6=    2    2    0    3    2    3    1
EVENT NO.    7=    2    2    1    3    3    2    0
EVENT NO.    8=    3    1    2    2    2    3    2
EVENT NO.    9=    3    2    1    2    3    2    2
EVENT NO.   10=    3    2    2    3    2    2    1
CLASS F( 1)
EVENT NO.    1=    2    0    2    3    3    3    2
EVENT NO.    2=    3    1    1    2    3    3    2
EVENT NO.    3=    3    1    2    3    2    3    1
EVENT NO.    4=    3    2    0    3    3    2    2
EVENT NO.    5=    3    2    1    3    3    3    1
EVENT NO.    6=    0    2    2    0    0    0    0
```

SYM4 Output for
Example 2 Section 4

Fig. 3

```
        CLASS FORMULI

F( 0) := (X2+X3+X7=3,5) & (X1+X4+X5+X6=10)
F( 1) := (X2+X3+X7=4) & (X1+X4+X5+X6=0,11,12)
```

CLASS COMPACT RATIO	MIN D GROUP RATIO	NUM. OF SYM VARS PER CLASS	CLASS
0.100	1.000	7	0
0.167	1.000	7	1

```
                    DISJOINT COVERS    DEFAULT SPECIFICATIONS    FACTOR VARS

SPACE ALLOCATED FOR G(F) IS  200                    SPACE ALLOCATED FOR MQ 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   SAVELQ=  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  4  3

NUMBER OF EVENTS SPECIFIED FOR EACH CLASS:
       CLASS    #EVENTS
         0         10
         1          6

                              **  AMOUNT OF UNUSED CORE =   8K  **


DLIST=     0    1

NCRIT=     2
CLIST=     1         2
TLIST=     0.00      0.00

LQ-STAR OPTION  LCST=   '1'B

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

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

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

Fig. 4

```
THE FOLLOWING  4 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS  0          COV  NEW  IND  TOT
COMPLEX:     (X1= 1 3 )   (X2= 0 1 )   (X4= 1 3 )   (X5= 1 3 )               4    4    4    4

COMPLEX:     (X1= 2 )   (X2= 1 2 )                                           3    3    1    7

COMPLEX:     (X3= 1 2 )   (X6= 2 )                                           3    2    2    9

COMPLEX:     (X3= 2 )   (X4= 2 )                                             2    1    1   10

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

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


THE FOLLOWING  5 CARTESIAN COMPLEXES COVER THE EVENTS IN CLASS  1          COV  NEW  IND  TOT
COMPLEX:     (X1= 2 )   (X2= 0 )   (X4= 3 )   (X6= 3 )                       1    1    1    1

COMPLEX:     (X1= 3 )   (X3= 1 )   (X4= 2 )   (X6= 3 )                       1    1    1    2

COMPLEX:     (X1= 0 3 )   (X4= 0 3 )   (X5= 0 2 )   (X6= 0 3 )               2    2    2    4

COMPLEX:     (X1= 3 )   (X2= 2 )   (X3= 0 )                                  1    1    1    5

COMPLEX:     (X1= 3 )   (X2= 2 )   (X3= 1 )   (X6= 3 )                       1    1    1    6

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

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

*** NORMAL TERMINATION ***
```

AQVAL/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

No. of Variables	Total No. of Events	No. of Classes	Total No. of Levels	'GEN'	Time	Region
3	35	4	12	No	(0,0.60)	116k
3	35	4	12	Yes	(0,0.62)	116k
7	16	2	21	No	(0,0.42)	116k
13	16	2	48	No	(0,0.75)	116k
35	260	15	100	No	(04,00)	200k
35	260	15	100	Yes	(30,00)	200k
50	287	3	130	No	(24,00)	250k

Figure 5

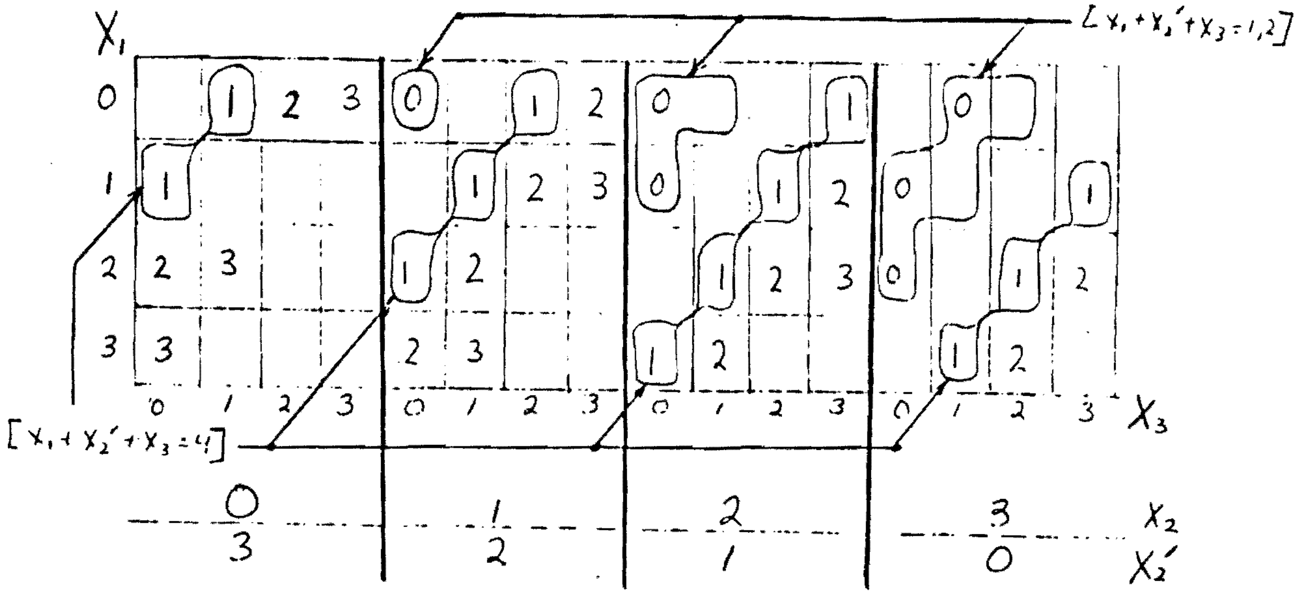
'GEN' requires considerably more time in problems with large numbers of variables. This effect should be noted in the specification of a time estimate for problems of this type.

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.

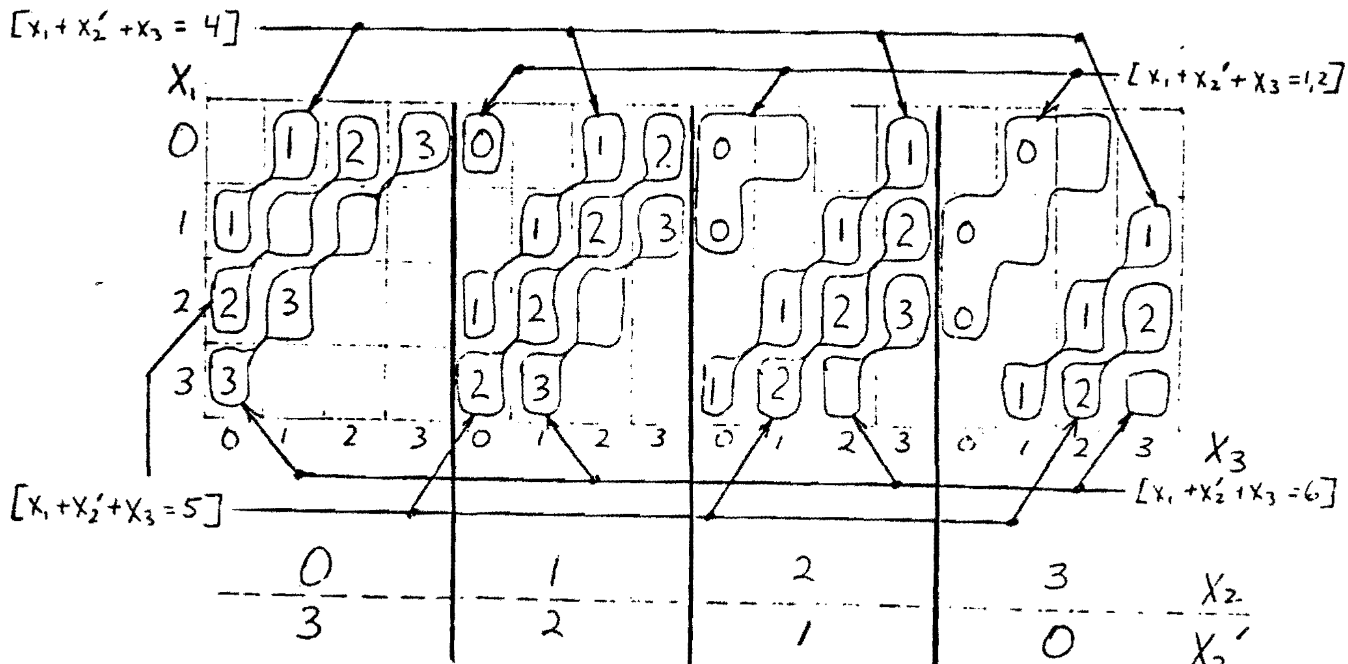
The GLD is a graphical model of the event space and complexes covering specific events. A complete description of GLDs can be found in paper[5]. Each cell of a GLD represents an event in the event space. The area(s) enclosed by curved lines represent events covered by selectors. In these particular GLDs, 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 Value	Parameter Name	Comment	Parameter Type

// EXEC FORTGCLG, REGION=118K
//SYSLIB DD DSN=USER.T2123.JENSEN76,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
			JCL

3 35 3 3	NV NROW MAXCL MAXD	Number of variables Total number of events Number of the last class Largest domain element	Storage Parameters
MAXPAT=100;	MAXPAT	Max. No. of events satisfying a given selector (Unspecified Parameters take default value)	Optional Parameters
0.25 4 4 4 6 12 11 6	CRITERIA1 DLIST(NV) NE(0:MAXCL)	User-specified compact ratio Domain sizes of the variables Number of events per class	Data Parameters
1 3 0 0 2 0 0 3 1 2 3 0 1 2 0 0 1 0 3 2 0 2 1 0 1 0 0 3 3 1 2 2 1 1 1 1 0 0 1 2 2 2 1 2 2 0 1 2 1 3 3 0 2 3 3 1 0 2 0 0 1 2 1 2 1 1 3 3 2 2 2 2 1 1 2 0 0 2 3 3 3 1 2 3 0 1 3 3 0 0 3 1 1 2 0 1 2 2 3 1 1 3 0 0 3	$\left.\right\}$ E(NROW,NV)	Event array Class 0 has 6 events Class 1 has 12 events Class 2 has 11 events Class 3 has 6 events	

Input for
Example 1A Section 5

Output for
Example 1A Section 5

```
GENERALIZE IS OFF
USERSPEC IS OFF
RERUN IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE    =   100
COMPACT_RATIO CRITERIA = 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         6
    1        12
    2        11
    3         6
CLASS F( 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 F( 1)
EVENT NO.     1=     3     2     0
EVENT NO.     2=     2     1     0
EVENT NO.     3=     1     0     0
EVENT NO.     4=     3     3     1
EVENT NO.     5=     2     2     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 F( 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 F( 3)
EVENT NO.     1=     3     0     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	CLASS COMPACT RATIO	MIN 0 GROUP RATIO	NUM. OF SYM VARS PER CLASS	CLASS
F(0) := (X1+X2'+X3=1,2)	0.667	0.500	3	0
F(1) := (X1+X2'+X3=4)	1.000	1.000	3	1
THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS 2			0	2
THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS 3			0	3

Parameter Value	Parameter Type

// EXEC [illegible]
//[illegible] DD DSN=[illegible],DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
	JCL

3 35 3 3	Storage Parameters
MAXPAT=100 GEN='1'B;	Optional Parameters
0.25 4 4 4 6 12 11 6 1 3 0 0 2 0 0 3 1 2 3 0 1 2 0 0 1 0 3 2 0 2 1 0 1 0 0 3 3 1 2 2 1 1 1 1 0 0 1 2 3 2 1 2 2 0 1 2 1 3 3 0 2 3 3 1 0 2 0 0 3 2 1 2 1 1 3 3 2 2 2 2 1 1 2 0 0 2 2 3 3 1 2 3 0 1 3 3 0 0 3 1 1 2 0 1 2 2 3 1 1 3 0 0 3	Data Parameters

Input for
Example 1B Section 5

```
GENERALIZE IS ON
USERSPEC IS OFF
RERUN IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE     =   100
COMPACT_RATIO CRITERIA = 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        6
   1       12
   2       11
   3        6
CLASS F( 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 F( 1)
EVENT NO.   1=    3    2    0
EVENT NO.   2=    2    1    0
EVENT NO.   3=    1    0    0
EVENT NO.   4=    3    3    1
EVENT NO.   5=    2    2    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 F( 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 F( 3)
EVENT NO.   1=    3    0    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
```

Output for
Example 1B Section 5

CLASS FORMULI	CLASS COMPACT RATIO	MIN D GROUP RATIO	NUM. OF SYM VARS PER CLASS	CLASS
F(0) := (X1+X2'+X3=1,2)	0.567	0.500	3	0
F(1) := (X1+X2'+X3=4)	1.000	1.000	3	1
F(2) := (X1+X2'+X3=5)	0.917	0.917	3	2
F(3) := (X1+X2'+X3=6)	0.600	0.600	3	3

Parameter Value	Parameter Name	Parameter Type

// EXEC PGM=JENSEN76, REGION=110K
//STEPLIB DD DSN=USER.P2123.JENSEN76,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
		JCL

7 16 1 3		Storage Parameters
USERSPEC=3;		Optional Parameters
0.5 1 3 3 4 4 4 3 10 6 1 0 1 3 3 3 0 0 0 2 2 3 3 1 3 1 0 3 2 2 0 1 1 1 1 2 3 1 1 1 2 0 1 0 2 2 2 0 1 0 0 1 2 2 1 2 0 1 2 2 1 2 2 0 3 0 3 2 1 1 0 2 0 3 2 2 0 0 0 1 0 0 2 3 2 2 0 0 1 1 1 2 2 0 1 1 2 3 3 2 1 1 2 0 0 0 3 0 2 2 1 3 0 0 1 2 2 2 0 2 2 2		Data Parameters
4 5 6	SPECLIST(1:USERSPED)	Data Parameters

Input for Example 2A Section 5

Parameter Value	Parameter Name	Parameter Type

// EXEC PGM=JENSEN76, REGION=110K
//STEPLIB DD DSN=USER.P2123.JENSEN76,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
		JCL

7 16 1 3		Storage Parameters
USERSPEC=3 RERUN='1'B;		Optional Parameters
0.5 4 3 3 4 4 4 3 10 6 1 0 1 3 3 3 0 0 0 2 2 3 3 1 3 1 0 3 2 2 0 1 1 1 1 2 3 1 1 1 2 0 1 0 2 2 2 0 1 0 0 1 2 2 1 2 0 1 2 2 1 2 2 0 3 0 3 2 1 1 0 2 0 3 2 2 0 0 0 1 0 0 2 3 2 2 0 0 1 1 1 2 2 0 1 1 2 3 3 2 1 1 2 0 0 0 3 0 2 2 1 3 0 0 1 2 2 2 0 2 2 2		Data Parameters
4 5 6	SPECLIST(1:USERSPEC)	Data Parameters

Input for Example 2B Section 5

```
GENERALIZE IS OFF
USERSPEC IS ON
USER SPECIFICATION LIST:  4  5  6
RERUN IS OFF
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE    =    500
COMPACT_RATIO CRITERIA = 0.50
NUMBER OF VARIABLES =       7
NUMBER OF EVENTS     =      16
NUMBER OF CLASSES    =       2
MAXIMUM LEVEL        =       3
NUMBER OF LEVELS FOR EACH VARIABLE :  4  3  3  4  4  4  3
NUMBER OF EVENT SPECIFIED FOR EACH CLASS :
CLASS    #EVENTS
   0         10
   1          6
CLASS F( 0)
EVENT NO.    1=    1    0    1    3    3    3    0
EVENT NO.    2=    0    0    2    2    3    3    1
EVENT NO.    3=    3    1    0    3    2    2    0
EVENT NO.    4=    1    1    1    1    2    3    1
EVENT NO.    5=    1    1    2    0    1    0    2
EVENT NO.    6=    2    2    0    1    0    0    1
EVENT NO.    7=    2    2    1    2    0    1    2
EVENT NO.    8=    2    1    2    2    0    3    0
EVENT NO.    9=    3    2    1    1    0    2    0
EVENT NO.   10=    3    2    2    0    0    0    1
CLASS F( 1)
EVENT NO.    1=    0    0    2    3    2    2    0
EVENT NO.    2=    0    1    1    1    2    2    0
EVENT NO.    3=    1    1    2    3    3    2    1
EVENT NO.    4=    1    2    0    0    0    3    0
EVENT NO.    5=    2    2    1    3    0    0    1
EVENT NO.    6=    2    2    2    0    2    2    2
```

Output for
Example 2A Section 5

```
                                                                              CLASS      MIN      NUM. OF
        CLASS FORMULI                                                        COMPACT   O GROUP   SYM VARS   CLASS
                                                                              RATIO     RATIO    PER CLASS

THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS  0                  0          0

THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS  1                  0          1
```

Output for
Example 2B Section 5

```
GENERALIZE IS OFF
USERSPEC IS ON
USER SPECIFICATION LIST:  4  5  6
RERUN IS ON
DATA REDUCTION IS OFF

MAXIMUM PATTERN SIZE    =   500
COMPACT_RATIO CRITERIA = 0.50
NUMBER OF VARIABLES =      7
NUMBER OF EVENTS     =     16
NUMBER OF CLASSES    =      2
MAXIMUM LEVEL        =      3
NUMBER OF LEVELS FOR EACH VARIABLE :  4   3   3   4   4   4   3
NUMBER OF EVENT SPECIFIED FOR EACH CLASS :
CLASS    #EVENTS
   0        10
   1         6
CLASS F( 0)
EVENT NO.    1=    1    0    1    3    3    3    0
EVENT NO.    2=    0    0    2    2    3    3    1
EVENT NO.    3=    3    1    0    3    2    2    0
EVENT NO.    4=    1    1    1    1    2    3    1
EVENT NO.    5=    1    1    2    0    1    0    2
EVENT NO.    6=    2    2    0    1    0    0    1
EVENT NO.    7=    2    2    1    2    0    1    2
EVENT NO.    8=    2    1    2    2    0    3    0
EVENT NO.    9=    3    2    1    1    0    2    0
EVENT NO.   10=    3    2    2    0    0    0    1
CLASS F( 1)
EVENT NO.    1=    0    0    2    3    2    2    0
EVENT NO.    2=    0    1    1    1    2    2    0
EVENT NO.    3=    1    1    2    3    3    2    1
EVENT NO.    4=    1    2    0    0    0    3    0
EVENT NO.    5=    2    2    1    3    0    0    1
EVENT NO.    6=    2    2    2    0    2    2    2

       CLASS FORMULA                                                CLASS      MIN      NUM. OF
                                                                   COMPACT  D GROUP   SYM VARS   CLASS
                                                                    RATIO    RATIO    PER CLASS

THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS  0        0        0

THERE ARE NO SYMMETRIC VARIABLES AND THEREFORE NO SYMMETRIC FORMULA FOR CLASS  1        0        1
THIS JOB IS RERUN WITH THE SAME EVENTS AND AUTO SELECTION OF D GROUPS.
______________________________________________________________________________
       CLASS FORMULA                                                CLASS      MIN      NUM. OF
                                                                   COMPACT  D GROUP   SYM VARS   CLASS
                                                                    RATIO    RATIO    PER CLASS

F( 0) := (X2+X3+X7'=3,5)                                           0.004    1.000        3        0

F( 1) := (X2+X3+X7'=4)                                             0.004    1.000        3        1
```

Parameter Value	Parameter Type

// EXEC PGM=JENSEN76, REGION=116K
//STEPLIB DD DSN=USER.P2123.JENSEN76,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
	JCL

6 17 2 4	Storage Parameters
DTRED='1'9)	Optional Parameters
0.5 3 4 4 3 5 4 6 4 7 0 3 0 1 0 0 1 1 1 0 4 1 0 2 0 0 3 1 0 1 0 0 3 2 1 0 1 0 1 2 0 2 1 1 2 0 2 0 0 1 2 2 1 2 0 2 3 0 2 1 1 1 0 0 1 0 1 2 1 1 0 3 6 2 4 0 2 2 2 0 4 2 1 1 2 1 3 3 2 3 2 2 2 1 2 3 1 0 1 2 0 1 3 2 1 2 1 0 3 1 0 3	Data Parameters

Input for
Example 3 Section 5

```
GENERALIZE IS OFF
USERSPEC IS OFF
PER IN IS OFF
DATA REDUCTION IS ON

MAXIMUM PATTERN SIZE     =    500
COMPACT_RATIO CRITERIA = 0.50
NUMBER OF VARIABLES =        6
NUMBER OF EVENTS       =   . 17
NUMBER OF CLASSES      =      3
MAXIMUM LEVEL              =      4
NUMBER OF LEVELS FOR EACH VARIABLE :   3   4   4   3   5   4
NUMBER OF EVENT SPECIFIED FOR EACH CLASS :
CLASS     #EVENTS
    0          6
    1          4
    2          7
CLASS F( 0)
EVENT NO.    1=    0    3    0    1    0    0
EVENT NO.    2=    1    1    1    0    4    1
EVENT NO.    3=    0    2    0    0    3    1
EVENT NO.    4=    0    1    0    0    3    2
EVENT NO.    5=    1    0    1    0    1    2
EVENT NO.    6=    0    2    1    1    2    0
CLASS F( 1)
EVENT NO.    1=    2    0    0    1    2    2
EVENT NO.    2=    1    2    0    2    3    0
EVENT NO.    3=    2    1    1    1    0    0
EVENT NO.    4=    1    0    1    2    1    1
CLASS F( 2)
EVENT NO.    1=    0    3    3    2    4    0
EVENT NO.    2=    2    2    2    0    4    2
EVENT NO.    3=    1    1    2    1    3    3
EVENT NO.    4=    2    3    2    2    2    1
EVENT NO.    5=    2    3    1    0    1    2
EVENT NO.    6=    0    1    3    2    1    2
EVENT NO.    7=    1    0    3    1    0    3

          CLASS FORMULI
```

Output for
Example 3 Section 5

```
                                           CLASS      MIN       NUM. OF
                                          COMPACT   D GROUP   SYM VARS   CLASS
                                           RATIO     RATIO    PER CLASS

F( 0) := (X2+X3+X6=3) & (X1+X4=0,1)        0.040     0.600        5        0

F( 1) := (X2+X3+X6=2) & (X1+X4=3)          0.067     0.667        5        1

F( 2) := (X2+X3+X6=6) & (X1+X4=2,4)        0.035     0.700        5        2


               *****DATA REDUCTION TABLE*****


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


                    X1   X2   X3   X4   X5   X6  |   X7   X8
CLASS F( 0)
EVENT NO.    1=     0    3    0    1    0    0  |   3    1
EVENT NO.    2=     1    1    1    0    4    1  |   3    1
EVENT NO.    3=     0    2    0    0    3    1  |   3    0
EVENT NO.    4=     0    1    0    0    3    2  |   3    0
EVENT NO.    5=     1    0    1    0    1    2  |   3    1
EVENT NO.    6=     0    2    1    1    2    0  |   3    1
CLASS F( 1)
EVENT NO.    1=     2    0    0    1    2    2  |   2    3
EVENT NO.    2=     1    2    0    2    3    0  |   2    3
EVENT NO.    3=     2    1    1    1    0    0  |   2    3
EVENT NO.    4=     1    0    1    2    1    1  |   2    3
CLASS F( 2)
EVENT NO.    1=     0    3    3    2    4    0  |   6    2
EVENT NO.    2=     2    2    2    0    4    2  |   6    2
EVENT NO.    3=     1    1    2    1    3    3  |   6    2
EVENT NO.    4=     2    3    2    2    2    1  |   6    4
EVENT NO.    5=     2    3    1    0    1    2  |   6    2
EVENT NO.    6=     0    1    3    2    1    2  |   6    2
EVENT NO.    7=     1    0    3    1    0    3  |   6    2
```

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 $\not{d}_1$*# 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 SYM4 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 VL_1 formulas containing both symmetric and non-symmetric selectors. The present version generates symmetric selectors only.

REFERENCES

1. Michalski, R.S., "VARIABLE-VALUED LOGIC: System VL_1," 1974 International Symposium on Multiple-valued Logic, West Virginia University, Morgantown, West Virginia, May 29-31, 1974.

2. Michalski, R.S., with James Larson: "AQVAL/1 (AQ7) User's Guide and Program Description," Report No. 731, Department of Computer Science, University of Illinois, Urbana, Illinois, June 1975.

3. User's Guide Section 1-01-10, Computing Services Office, University of Illinois at Urbana-Champaign, February 1972.

4. Michalski, R.S., "Learning by Inductive Inference," NATO Advanced Study Institute on Computer-oriented Learning Processes, Aug. 26-Sept.7, 1974, France (invited paper)

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

6. Kohavi, Zvi, Switching and Finite Automata Theory McGraw-Hill, Inc., New York, 1970 Pages 158--171

7. Michalski, R.S., "AQVAL/1--Computer Implementation of a Variable-Valued Logic System and the Application to Pattern Recognition," Proceedings of the First International Joint Conference on Pattern Recognition, Washington, D.C., October 30-November 1, 1973.

8. Jensen, Gerald M., "SYM-1: A Program That Detects Symmetry of Variable-Valued Logic Functions", May, 1975, Department of Computer Science, University of Illinois, Urbana, Illinois.

APPENDIX A

SOURCE LISTING OF SYM4

```
PL/I OPTIMIZING COMPILER          MAINP:    PROC OPTIONS(MAIN) REORDER;

   STMT LEV NT

     2   1   0  |          DCL (NV,NROW,MAXCL,MAXO) FIXED BIN;                          |
     3   1   0  |          ON ENDFILE(SYSIN) GOTO EXIT;                                 |
     4   1   0  |          OPEN FILE(SYSPRINT) PAGESIZE(66);                            |
     5   1   0  |NEXT:     GET FILE(SYSIN) LIST(NV,NROW,MAXCL,MAXO);                    |
     6   1   0  |          BEGIN;                                                       |
     7   2   0  |          DCL (I,IV1,IY,I2,I3,I4,J,J2,J3,K,                            |
                |          NE(0:MAXCL),ROWCL(0:MAXCL),E(NROW,NV),ICL,IO,IROW,ROW1,ROW2, |
                |          COLSUM,N,D,IV,IV2,#ROWS,NO,TOP,TOP2,IBUF,IRS,PT1,PT2,NCL,    |
                |          OLIST(NV),DLINK(NV),DTOP(NV),DTOP#(NV),$TOP,$#COLS,$LIST(NV),|
                |          MAXTOP,MAX#,MAXLINK(NV),#REPETITIONS,DSUBTOP#(NV),#RS,CL,DL, |
                |          SDMAXLINK(NV),SDSUBTOP(NV),SDSUBTOP#(NV),S#DSUBTOP,SLC,MIN,  |
                |          EV(NROW,NV) INIT((NROW*NV) 0B),DV(NROW),SPECLIST(NV),ICOL2,  |
                |          DMAXLINK(NV),DSUBTOP(NV),#DSUBTOP,DMAX#,DCS(NV),MAXPAT,COUNT,|
                |          USERSPEC,#CS,CS_TOP(NV),CS#(NV),CS(NV),CS_LINK(NV)) FIXED BIN,|
                |          (COMPL(NV),SCOMPL(NV),INCOMPL(0B:MAXO*NV),NOTCOMPACTB,GEN,   |
                |          RERUN,DTRED) BIT(1) ALIGNED,                                 |
                |          (CRITERIAL,RS#(0B:MAXO*NV),#COMP,RATIO,MINRATIO)             |
                |          FLOAT BIN(53),RETLOC LABEL,                                  |
                |          NOF CHAR(10) ALIGNED INIT('NUMBER OF '),                     |
                |          CH2(10B) CHAR(1) ALIGNED INIT( ' ',',',' ),                  |
                |          (ISAVE,CPBIN) FIXED BIN(31);                                 |

     8   2   0  |          MAXPAT=500;       /*  DEFAULT VALUE  */                      |
     9   2   0  |          GEN='0'B;         /*  DEFAULT VALUE  */                      |
    10   2   0  |          USERSPEC=0B;    /*  DEFAULT VALUE  */                        |
    11   2   0  |          RERUN='0'B;       /*  DEFAULT VALUE  */                      |
    12   2   0  |          DTRED='0'B;                                                  |
    13   2   0  |          GET FILE(SYSIN) DATA(MAXPAT,GEN,USERSPEC,RERUN,DTRED);       |

    14   2   0  |          GET FILE(SYSIN) LIST(CRITERIAL,OLIST,NE,E);                  |
    15   2   0  |          IF USERSPEC=0B THEN RERUN='0'B;                              |
    16   2   0  |                            ELSE GET FILE(SYSIN) LIST ((SPECLIST(I)    |
                |                                           DO I=1B TO USERSPEC));      |
    17   2   0  |          NCL=MAXCL+1B;                                                |
                |/*  PUT ROW_DISPLACEMENT OF EACH EVENT_CLASS INTO ROWCL             */|
    18   2   0  |          ROWCL(0B)=0B;                                                |
    19   2   0  |          DO ICL=0B TO MAXCL-1B;                                       |
    20   2   1  |             ROWCL(ICL+1B)=ROWCL(ICL)+NE(ICL);                         |
    21   2   1  |          END;                                                         |

                |/*    ECHO THE INPUT DATA                                   */        |
    22   2   0  |          IF GEN THEN PUT PAGE FILE(SYSPRINT) EDIT ('GENERALIZE IS ON') |
    23   2   0  |                  (A); ELSE PUT PAGE FILE(SYSPRINT) EDIT ('GENERALIZE', |
                |                  ' IS OFF')(A,A);                                     |
    24   2   0  |          IF USERSPEC¬=0B THEN PUT SKIP FILE(SYSPRINT) EDIT ('USERSPEC',|
                |                  ' IS ON','USER SPECIFICATION LIST:',(SPECLIST(I)     |
                |                     DO I=1B TO USERSPEC))(A,A,SKIP,A,(USERSPEC)       |
```

```
PL/I OPTIMIZING COMPILER        MAINP:   PROC OPTIONS(MAIN) REORDER;

   STMT LEV NT

     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')     |
     27    2  0  |                  (A); ELSE PUT SKIP FILE(SYSPRINT) EDIT ('RERUN',        |
                 |                  ' IS OFF')(A,A);                                          |
     28    2  0  |            IF DTRED THEN PUT SKIP FILE(SYSPRINT) EDIT ('DATA RE',        |
     29    2  0  |                  'DUCTION 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       =',MAXD,                                  |
                 |            NOF,'LEVELS FOR EACH VARIABLE :',DLIST,                        |
                 |            NOF,'EVENT SPECIFIED FOR EACH CLASS :','CLASS   #EVENTS',     |
                 |            (ICL,NE(ICL) DO ICL=0B TO MAXCL),                              |
                 |            ('CLASS F(',ICL,')',                                           |
                 |              ('EVENT NO.',IROW-ROWCL(ICL),'=',E(IROW,*)                   |
                 |              DO IROW=ROWCL(ICL)+1B TO ROWCL(ICL)+NE(ICL))                 |
                 |            DO ICL=0B 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,2 A,SKIP,A,(NCL)(SKIP,F(5),F(10)),             |
                 |            (NCL)(SKIP,A,F(2),A,(NE(ICL))(SKIP,A,F(5),A,(NV)(F(5)))));    |

                 |/*  NOW CHECK FOR DISJOINT EVENT CLASSES.                               */|

     31    2  0  |            DO I2=0B TO MAXCL-1B;                                          |
     32    2  1  |            ROW1=ROWCL(I2)+1B;                                             |
     33    2  1  |            ROW2=ROW1+NE(I2)-1B;                                           |
     34    2  1  |               DO I3=ROW1 TO ROW2;                                         |
     35    2  2  |                   DO J2=(ROW2+1) TO NROW;                                 |
     36    2  3  |                      DO J3=1B TO NV;                                      |
     37    2  4  |                         IF E(I3,J3) ¬= E(J2,J3) THEN GO TO OK;            |
     38    2  4  |                      END;                                                 |
     39    2  3  |                      DO I4=(I2+1B) TO MAXCL;                              |
     40    2  4  |                         IF ROWCL(I4) >= J2 THEN GOTO OUT;                 |
     41    2  4  |                      END;                                                 |
     42    2  3  |OUT:                  I4=I4-1B;                                            |
     43    2  3  |                      PUT SKIP FILE(SYSPRINT) EDIT                         |
                 |                      ('THE PROGRAM OPERATES ON DISJOINT EVENT CLASSES.',  |
                 |                      'EVENT NO. ',(I3-ROWCL(I2)),' IN CLASS NO. ',I2,     |
                 |                      ' IS THE SAME AS EVENT NO. ',(J2-ROWCL(I4)),         |
                 |                      ' IN CLASS NO. ',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        MAINP:   PROC OPTIONS(MAIN) RECRDER;

   STMT LEV NT

     44   2  3  |                        GOTO NEXT;
     45   2  3  |OK:                 END;
     46   2  2  |                 END;
     47   2  1  |              END;

                |      /*   IF THE USER DOES NCT 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 THOSES
                |           SPECIFIED WILL BE CONSIDERED FOR SYMMETRY       */

     48   2  0  |           IF USERSPEC=0B THEN GOTO LINKER;
     49   2  0  |           ND=0B; DLINK,DTOP,DTOP#=0B;
     51   2  0  |           DO IV=USERSPEC TO 1B BY -1B;
     52   2  1  |              DO ID=1B TO ND;
     53   2  2  |                 IF DLIST(SPECLIST(IV))=DLIST(DTOP(ID)) THEN GOTO PUSHD1;
     54   2  2  |              END;
     55   2  1  |NEWD1:     ND=ND+1B;
     56   2  1  |PUSHD1:    DTOP#(ID)=DTOP#(ID)+1B;
     57   2  1  |           DLINK(IV)=DTOP(ID);
     58   2  1  |           DTOP(ID)=SPECLIST(IV);
     59   2  1  |           END;
     60   2  0  |           DO ID=1B TO ND;
     61   2  1  |              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 ',
                |                                    'SPECLIST.')(A,A,F(3),A,A,A);
     63   2  2  |                                    GOTO TWO;
     64   2  2  |                              END;
     65   2  1  |TWO:       END;
     66   2  0  |           GOTO START;

                |/* SINCE THERE IS NO USER SPECIFICATION OF VARIABLES,                  */
                |/*  PARTITION ALL VARIABLES INTO 'D_GROUPS' WITH SAME #LEVELS          */
                |/*    & LINK THE 'D_GROUPS' BY DLINK                                    */
     67   2  0  |LINKER:  ND=0B; DLINK,DTOP,DTOP#=0B; RERUN='0'B;
     70   2  0  |           DO IV=NV TO 1B BY -1B;
     71   2  1  |              DO ID=1B TO ND;
     72   2  2  |                 IF DLIST(IV)=DLIST(DTOP(ID)) THEN GOTO PUSHD;
     73   2  2  |              END;
     74   2  1  |NEWD:      ND=ND+1B;
     75   2  1  |PUSHD:     DTOP#(ID)=DTOP#(ID)+1B;
     76   2  1  |           DLINK(IV)=DTOP(ID);
     77   2  1  |           DTOP(ID)=IV;
     78   2  1  |           END;
     79   2  0  |START:     DLIST=DLIST-1B;
```

```
    STMT LEV NT

                  |/*  FIND SYMMETRY OF VARIABLES IN EACH CLASS                               */|
      80   2  0   |          DO ICL=0B TO MAXCL;                                                 |

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

      81   2  1   |            #DSUBTOP,DMAX#=0B; DMAXLINK,DSUBTOP,DSUBTOP#=0B;                  |
      83   2  1   |            ROW1=ROWCL(ICL)+1B;                                               |
      84   2  1   |            #ROWS=NE(ICL);                                                    |
      85   2  1   |            ROW2=ROW1+#ROWS-1B;                                               |
      86   2  1   |            COMPL='0'B;                                                       |
      87   2  1   |            DO IV=1 TO NV;                                                    |
      88   2  2   |            COLSUM=0B;                                                        |
      89   2  2   |            DO IROW=ROW1 TO ROW2;                                             |
      90   2  3   |               COLSUM=COLSUM+E(IROW,IV);                                      |
      91   2  3   |            END;                                                              |
      92   2  2   |            DCS(IV)=COLSUM;                                                   |
      93   2  2   |            END;                                                              |

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

      94   2  1   |            DO ID=1B TO NO;                                                   |
                  |            /*  IGNORE 'D_GROUP' WITH ONLY 1 VARIABLE  */                     |
      95   2  2   |               IF DTOP#(ID)=1B THEN GOTO NEXTD;                               |
      96   2  2   |               N=DTOP#(ID);                                                   |
      97   2  2   |               D=DLIST(DTOP(ID));                                             |

                  |/*  FIND 'TOTAL SYMMETRY' OF ALL VARIABLES IN THE 'D_GROUP'                 |
                  |    COMPLEMENT APPROPRIATE VARIABLES.                                       |
                  |    THE VALUE OF THE SWITCH GEN DETERMINES WHICH COMPLIMENT 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.       |
                  |    (IDEALLY ONE ROW SUM)                                                 */|

      98   2  2   |TOTALSORT:                                                                  |
                  |         IF GEN='0'B THEN GOTO OTHRN;                                       |
```

```
PL/I OPTIMIZING COMPILER         MAINP:   PROC OPTIONS(MAIN) RECRDER;

    STMT LEV NT

     99   2  2 |          IF N>11111B THEN DO; PUT SKIP FILE(SYSPRINT) EDIT          |
               |                            ('THE GENERALIZATION ROUTINE ALLOWS A MAX',  |
               |                            'IMUM OF 31 VARIABLES IN ANY D GROUP.',     |
               |                            '  THIS JOB IS NOT GENERALIZED.')           |
               |                            (A,A,A);                                     |
    101   2  3 |                            GOTO OTHRN;                                  |
    102   2  3 |                            END;                                         |
    103   2  2 |          CPBIN=0B;                                                      |
    104   2  2 |          MIN=NE(ICL);                                                   |
    105   2  2 |          IF NE(ICL)=1B THEN GOTO OTHRN;                                 |

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

```
STMT LEV NT

 139   2  3  |              IV=OLINK(IV);                                                |
 140   2  3  |           END;                                                            |
             |/*  COMPLEMENT EVENTS USING THE PROPER COMPLEMENT VECTOR               */ |
 141   2  2  |OUT1:      IV=DTOP(ID);                                                    |
 142   2  2  |            DO WHILE (IV>0B);                                              |
 143   2  3  |               IF COMPL(IV)='0'B THEN GOTO NEXTV3;                         |
 144   2  3  |                  DO IROW=ROW1 TO ROW2;                                    |
 145   2  4  |                     E(IROW,IV)=D-E(IROW,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;                                                   |

             |/*  IN THIS SECOND ROUTINE, THE FACT THAT THE SYMMETRY IS COMPLETELY      |
             |    SPECIFIED, IS UTILIZED.  IF THE SYMMETRY IS COMPLETELY SPECIFIED,     |
             |    THE VARS WILL ALSO HAVE THE PROPERTY OF THE SAME COLUMN SUM IN        |
             |    THE EVENT CLASS. THEREFORE THE VARS ARE PARTITIONED INTO 'D_GROUPS'   |
             |    & IF NECESSARY FURTHER PARTITIONED INTO 'CS_GROUPS'.                  |
             |    IN FORMING 'CS_GROUPS':                                               |
             |    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  |OTHRN:       IV=DTOP(ID);                                                  |
 152   2  2  |             DO WHILE(IV>0B);                                              |
 153   2  3  |                IF COMPL(IV) THEN GOTO NEXTV;                              |
 154   2  3  |                IV2=OLINK(IV);                                             |
 155   2  3  |                DO WHILE(IV2>0B);                                          |
 156   2  4  |                   IF COMPL(IV2) THEN GOTO NEXTV2;                         |
 157   2  4  |                   IF DCS(IV) ¬= D**ROWS-DCS(IV2) THEN GOTO NEXTV2;        |
 158   2  4  |                   IF DCS(IV)  = DCS(IV2) THEN GOTO NEXTV2;                |
 159   2  4  |                   COMPL(IV2)='1'B;                                        |
 160   2  4  |                   DO IROW=ROW1 TO ROW2;                                   |
 161   2  5  |                      E(IROW,IV2)=D-E(IROW,IV2);                           |
 162   2  5  |                   END;                                                    |
 163   2  4  |                   DCS(IV2)=DCS(IV);                                       |
 164   2  4  |NEXTV2:         IV2=OLINK(IV2); END;                                       |
 166   2  3  |NEXTV:       IV=OLINK(IV); END;                                            |
             |/* TEST ENTIRE D_GROUP FOR TOTAL SYMMETRY                              */ |
 168   2  2  |TESTPT:  IF SUF(DTOP(ID),OLINK,N) THEN GOTO VER;                         |
 169   2  2  |         IF NOTCOMPACTB THEN GOTO SORTCS;                                  |
 170   2  2  |VER:     IF VERIFY(DTOP(ID),OLINK,N) THEN GOTO TOTALSYM;                   |
 171   2  2  |         GOTO SORTCS;                                                      |


             |/*  ALL VARIABLES IN 'D_GROUP' ARE SYMMETRIC                           */ |
 172   2  2  |TOTALSYM:   MAXTOP=DTOP(ID);                                              |
 173   2  2  |            MAX#=N;                                                        |
```

```
STMT LEV NT

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

             |/*  FURTHER PARTITION  VARIABLES OF 'D_GROUP' INTO 'CS_GROUPS'          */|
             |/*                       WITH SAME COLUMN SUM                           */|
             |/*  SORT 'CS_GROUPS' IN DESCENDING ORDER OF THEIR SIZES,                  |
             |     I.E.  THE NUMBER OF VARIABLES THEY CONTAIN                         */|

 177   2  2  |SORTCS:  #CS=0B; CS_TOP,CS#,CS_LINK=0B; CS=-1B;                          |
 180   2  2  |         IV=DTOP(ID);                                                     |
 181   2  2  |         DO WHILE(IV>0B);                                                 |
 182   2  3  |LINKCS:  DO TOP=1B TO #CS;                                                |
 183   2  4  |         IF CS(TOP) = DCS(IV) THEN GOTO UPDATE;                           |
 184   2  4  |         END;                                                             |
 185   2  3  |NEWCS:   #CS=#CS+1B;                                                      |
 186   2  3  |         CS(#CS)=DCS(IV);                                                 |
 187   2  3  |UPDATE:  CS#(TOP)=CS#(TOP)+1B;                                            |
 188   2  3  |         CS_LINK(IV)=CS_TOP(TOP);                                         |
 189   2  3  |         CS_TOP(TOP)=IV;                                                  |
 190   2  3  |         IV=DLINK(IV);                                                    |
 191   2  3  |         END;                                                             |
 192   2  2  |         IF #CS>1B THEN                                                   |
             |TOPSORT: DO TOP=1B TO #CS-1B;                                             |
 193   2  3  |            DO TOP2=TOP+1B TO #CS;                                        |
 194   2  4  |            IF CS#(TOP)>=CS#(TOP2) THEN GOTO NEXT_TOP2;                   |
 195   2  4  |            IBUF=CS(TOP); CS(TOP)=CS(TOP2); CS(TOP2)=IBUF;                |
 198   2  4  |            IBUF=CS#(TOP); CS#(TOP)=CS#(TOP2); CS#(TOP2)=IBUF;            |
 201   2  4  |            IBUF=CS_TOP(TOP); CS_TOP(TOP)=CS_TOP(TOP2);                   |
 203   2  4  |                                            CS_TOP(TOP2)=IBUF;            |
 204   2  4  |NEXT_TOP2:  END;                                                          |
 205   2  3  |         END TOPSORT;                                                     |

             |/*  FIND 'TOTAL OR PARTIAL'SYMMETRY OF ALL 'CS_GROUPS',                   |
             |    IGNORE THOSE WITH ONLY 1 VARIABLE                                     |
             |    & GOTO PROCESS NEXT 'D_GROUP'.                                      */|

 206   2  2  |         TOP=0B;                                                          |
 207   2  2  |NEXTICS: TOP=TOP+1B;                                                      |
 208   2  2  |         IF CS#(TOP)¬>1B THEN GOTO NEXTD;                                 |
 209   2  2  |            IF SUF(CS_TOP(TOP),CS_LINK,CS#(TOP)) THEN GOTO VER2;          |
 210   2  2  |            IF NOTCOMPACTB THEN GOTO PSYM;                                |
 211   2  2  |VER2:        IF VERIFY(CS_TOP(TOP),CS_LINK,CS#(TOP)) THEN GOTO TSYM;       |
 212   2  2  |PSYM:    $TOP=CS_TOP(TOP); $LIST=CS_LINK; $#COLS=CS#(TOP);                |
 215   2  2  |NEXTJ:   MAXTOP,MAX#=0B; MAXLINK=0B;                                      |
 217   2  2  |         J,K=$TOP; PT2=$LIST(J); GOTO ADDJ;                               |
 220   2  2  |COMPARE: PT2=$LIST(K);                                                    |
 221   2  2  |         IF ¬SUF2(J,K) THEN PT1=K; ELSE                                   |
```

```
PL/I OPTIMIZING COMPILER          MAINP:    PROC OPTIONS(MAIN) REORDER;

    STMT LEV NT

     222   2  2  |ADDJ:                DO;                                                            |
     223   2  3  |                        $#CCLS=$#CCLS-1B;                                           |
     224   2  3  |                        IF $TOP=K THEN $TOP=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>0B 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 NOTCOMPACTB 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 $#COLS>1B THEN GOTO NEXTJ;                                         |
     240   2  2  |              GOTO NEXTICS;                                                         |

                 |/*  ALL VARIABLES IN 'CS_GROUP' ARE SYMMETRIC                                  */|
     241   2  2  |TSYM:         MAXTOP=CS_TOP(TOP);                                                   |
     242   2  2  |              MAX#=CS#(TOP);                                                        |
     243   2  2  |              MAXLINK=CS_LINK;                                                      |
     244   2  2  |              RETLOC=NEXTICS;                                                       |
                 |/*  MERGE & LINK SYMMETRIC VARIABLES IN DMAXLINK                               */|
     245   2  2  |LINK:         #DSUBTOP=#DSUBTOP+1B;                                                 |
     246   2  2  |              PT1,DSUBTOP(#DSUBTOP)=MAXTOP;                                         |
     247   2  2  |              DSUBTOP#(#DSUBTOP)=MAX#;                                              |
     248   2  2  |              DO WHILE(PT1>0B);                                                     |
     249   2  3  |                 PT2=PT1;                                                           |
     250   2  3  |                 PT1=MAXLINK(PT1);                                                  |
     251   2  3  |                 DMAXLINK(PT2)=PT1;                                                 |
     252   2  3  |              END;                                                                  |
     253   2  2  |              DMAX#=DMAX#+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  |VERIFY:  PROC(NODE1PT,LIST,#COLS) RETURNS(BIT(1));                              |
     256   3  2  |              DCL (NODE1PT,LIST(*),#CCLS,ICL2,KRS,KROW) FIXED BIN;                  |
                 |                                                                                    |
     257   3  2  |              DO ICL2=0B TO ICL-1B, ICL+1B TO MAXCL;                                |
     258   3  3  |                 DO IROW=1B TO NE(ICL2);                                            |
```

```
PL/I OPTIMIZING COMPILER          MAINP:    PROC OPTIONS(MAIN) REORDER;

   STMT LEV NT

    259    3   4  |                  KRS=0B; KRCW=IROW+ROWCL(ICL2);                          |
    261    3   4  |                  IV=NODE1PT;                                             |
    262    3   4  |                  DO WHILE(IV>0B);                                        |
    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 INCOMPL(KRS) THEN RETURN('0'B);                      |
    268    3   4  |                  IF RS#(KRS)¬=0B THEN RETURN('0'B);                      |
    269    3   4  |               END;                                                       |
    270    3   3  |            END;                                                          |
    271    3   2  |            INCOMPL='0'B; RETURN('1'B);                                   |
    273    3   2  |END VERIFY;                                                               |

                  |/*  'SUF2' FINDS PAIRWISE SYMMETRY OF 2 VARIABLES J & K                */|
    274    2   2  |SUF2:      PROC(J,K) RETURNS(BIT(1));                                     |
    275    3   2  |            RS#=0E0B;                                                     |
    276    3   2  |            DO IROW=ROW1 TO ROW2;                                         |
    277    3   3  |               IRS=E(IROW,J)+E(IROW,K);                                   |
    278    3   3  |               RS#(IRS)=RS#(IRS)+1E0B;                                    |
    279    3   3  |            END;                                                          |

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

    280    3   2  |            DO IRS=0B TO D*10B;                                           |
    281    3   3  |               IF RS#(IRS) = 0E0B THEN GOTO NEXT_RS;                      |
    282    3   3  |               #COMP=COMP(10B,IRS,D);                                     |
    283    3   3  |               IF #ROWS/#COMP < CRITERIAL                                 |
                  |                  /*  NO NEED TO CHECK THIS GROUP OF VARIABLES            |
                  |                      SINCE COMPACT_RATIO CRITERIAL IS NOT SATISFIED   */|
                  |                  THEN RETURN('0'B);                                      |
    284    3   3  |            IF #COMP>MAXPAT                                               |
                  |               THEN  /*  PATTERN SIZE EXCEEDS MAXPAT   */                 |
                  |                   DO;                                                    |
    285    3   4  |                      PUT FILE(SYSPRINT) EDIT                             |
                  |                      ('PATTERN SIZE=',#COMP,' EXCEEDS MAXPAT=',MAXPAT)   |
                  |                      (SKIP,2 (A,F(11)));                                 |
    286    3   4  |                      RETURN('0'B);                                       |
    287    3   4  |                  END;                                                    |
    288    3   3  |               BEGIN;                                                     |
    289    4   3  |               DCL (LISTRS(#COMP,10B),LEVEL(10B),ICOMP,ICOL)              |
                  |               FIXED BIN;                                                 |
                  |/*  CALL 'RSLIST' TO GENERATE ALL POSSIBLE PATTERNS.                    */|
    290    4   3  |               CALL RSLIST(10B,IRS,D,LISTRS,LEVEL);                       |
    291    4   3  |               DO ICOMP=1B TO #COMP;                                      |
    292    4   4  |                  DO IROW=ROW1 TO ROW2;                                   |
    293    4   5  |                     DO ICOL=1B TO 10B;                                   |
```

```
STMT LEV NT

 294   4   6  |                      IF ICOL=1B THEN IV=J; ELSE IV=K;            |
 296   4   6  |                      IF E(IROW,IV)¬=LISTRS(ICOMP,ICOL)            |
              |                         THEN GOTO NEXT_ROW;                       |
 297   4   6  |                   END;                                            |
 298   4   5  |                   /*  MATCH THIS PATTERN  */  GOTO NEXT_PATTERN;  |
 299   4   5  |NEXT_ROW:       END;                                               |
              |                /*  NOT MATCH THIS PATTERN  */                     |
 300   4   4  |                RETURN('0'B);                                      |
 301   4   4  |NEXT_PATTERN:   END;                                               |
              |                /*  MATCH ALL PATTERNS  */                         |
 302   4   3  |             END;                                                  |
 303   3   3  |NEXT_RS: END;                                                      |
 304   3   2  |          RETURN('1'B);                                            |
 305   3   2  |END SUF2;                                                          |

              |/*  'SUF'  TESTS 'TOTAL SYMMETRY' OF A LIST OF VARIABLES          |
              |    1.  #OCCURRENCES OF EXISTING 'ROW_SUMS' ARE COLLECTED          |
              |    2. THE VARIABLES ARE 'TOTALLY 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_RATIO' 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  |SUF:      PROC (NODE1PT,LIST,#COLS) RETURNS(BIT(1));               |
 307   3   2  |          DCL (NODE1PT,LIST(*),#COLS) FIXED BIN,                   |
              |          SUFBIT BIT(1) ALIGNED INIT('1'B);                        |
 308   3   2  |          NOTCOMPACTB='0'B; INCOMPL='0'B; RS#=0E0B;                |
 311   3   2  |          DO IROW=ROW1 TO ROW2;                                    |
 312   3   3  |             IV=NODE1PT; IRS=0B;                                   |
 314   3   3  |             DO WHILE(IV>0B);                                      |
 315   3   4  |                IRS=IRS+E(IROW,IV);                                |
 316   3   4  |                IV=LIST(IV);                                       |
 317   3   4  |             END;                                                  |
 318   3   3  |             RS#(IRS)=RS#(IRS)+1E0B;                               |
 319   3   3  |          END;                                                     |

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

 320   3   2  |SUFRS:    DO IRS=0B TO 0*#COLS;                                    |
 321   3   3  |             IF RS#(IRS) = 0E0B THEN GOTO NEXT_RS;                 |
 322   3   3  |             #COMP=COMP(#COLS,IRS,0);                              |
 323   3   3  |             IF #ROWS/#COMP < CRITERIA1                            |
              |                THEN                                               |
              |                /*  NO NEED TO CHECK THIS GROUP OF VARIABLES       |
```

```
    STMT LEV NT

                    |                 SINCE COMPACT_RATIO CRITERIAL IS NOT SATISFIED     */|
                    |             DO;                                                      |
     324   3   4    |                NOTCOMPACTB='1'B;  RETURN('0'B);                      |
     326   3   4    |             END;                                                     |
     327   3   3    |       IF #CCMP>MAXPAT                                                |
                    |          THEN  /*  PATTERN SIZE EXCEEDS MAXPAT  */                   |
                    |              DO;                                                     |
     328   3   4    |                 NOTCOMPACTB='1'B;                                    |
     329   3   4    |                 PUT FILE(SYSPRINT) EDIT                              |
                    |                 ('PATTERN SIZE=',#COMP,' EXCEEDS MAXPAT=',MAXPAT)    |
                    |                 (SKIP,2 (A,F(11)));                                  |
     330   3   4    |                 RETURN('0'B);                                        |
     331   3   4    |             END;                                                     |
     332   3   3    |          BEGIN;                                                      |
     333   4   3    |          DCL (LISTRS(#COMP,#COLS),LEVEL(#COLS),ICOMP,ICOL,NOTOCCUR#)|
                    |          FIXED BIN;                                                  |
     334   4   3    |          NOTOCCUR#=0B;                                               |
                    |/*  CALL 'RSLIST' TO GENERATE ALL POSSIBLE PATTERNS.               */|
     335   4   3    |          CALL RSLIST(#COLS,IRS,0,LISTRS,LEVEL);                      |
     336   4   3    |          DO ICOMP=1B TO #COMP;  /*  CHECK ALL PATTERNS  */           |
     337   4   4    |             DO IROW=ROW1 TO ROW2;  /*  AGAINST ALL EVENTS IN          |
                    |                                         CLASS.                   */ |
     338   4   5    |             IV=NODELPT;                                              |
     339   4   5    |             DO ICOL=1B TO #COLS;                                     |
     340   4   6    |                IF E(IROW,IV)¬=LISTRS(ICOMP,ICOL) THEN GOTO NEXT_ROW;|
     341   4   6    |                IV=LIST(IV);                                          |
     342   4   6    |             END;                                                     |
     343   4   5    |             /*  MATCH THIS PATTERN  */  GOTO NEXT_PATTERN;           |
     344   4   5    |NEXT_ROW:    END;                                                     |
                    |             /* NOT MATCH THIS PATTERN  */                            |
     345   4   4    |             SUFBIT='0'B;                                             |
     346   4   4    |             INCOMPL(IRS)='1'B;                                       |
     347   4   4    |             NOTOCCUR#=NOTOCCUR#+1B;                                  |
     348   4   4    |             RATIO=(#CCMP-NOTOCCUR#)/#COMP;                           |
     349   4   4    |             IF RATIO>=CRITERIAL THEN GOTO NEXT_PATTERN;              |
     350   4   4    |             NOTCOMPACTB='1'B;  RETURN('0'B);                         |
     352   4   4    |NEXT_PATTERN:                                                         |
                    |          END;                                                        |
                    |          /*  MATCH ALL PATTERNS  */                                  |
     353   4   3    |          END;                                                        |
     354   3   3    |NEXT_RS: END;                                                         |


     355   3   2    |       RETURN(SUFBIT);                                                |
     356   3   2    |END SUF;                                                              |

                    |/*                                                                    |
                    |'RSLIST'  GENERATES ALL POSSIBLE PATTERNS                             |
```

```
PL/I OPTIMIZING COMPILER        MAINP:   PROC OPTIONS(MAIN) REORDER;

    STMT LEV NT

                   |FOR N VARIABLES, WITH SAME MAXIMUM LEVEL -- D,                              |
                   |TO SUM TO IRS -- 'ROW_SUM' & PUT THEM IN LISTRS(*,*) .                     */|
     357   2   2   |RSLIST:  PROC(N,IRS,D,LISTRS,LEVEL);                                        |
     358   3   2   |         DCL (LISTRS(*,*),LEVEL(*)) FIXED BIN;                              |
     359   3   2   |         DCL (IROW,N,IRS,D,NV,R,IV) FIXED BIN;                              |
     360   3   2   |         IROW,NV=1B;                                                        |
     361   3   2   |NEXTV:   IF N=NV THEN DO; R=IRS;                                            |
     363   3   3   |                           DO IV=1B TO N-1B;                                |
     364   3   4   |                              R=R-LEVEL(IV);                                |
     365   3   4   |                           END;                                             |
     366   3   3   |                           IF R>D    | R<0B THEN GOTO POP;                  |
     367   3   3   |                           LEVEL(N)=R;                                      |
     368   3   3   |                           LISTRS(IROW,*)=LEVEL;                            |
     369   3   3   |                           IROW=IROW+1B;                                    |
     370   3   3   |                           GOTO POP;                                        |
     371   3   3   |                      END;                                                  |
     372   3   2   |         LEVEL(NV)=0B;                                                      |
     373   3   2   |PUSH:    NV=NV+1B;                                                          |
     374   3   2   |         GOTO NEXTV;                                                        |
     375   3   2   |POP:     NV=NV-1B;                                                          |
     376   3   2   |         IF NV=0B THEN RETURN;                                              |
     377   3   2   |         IF LEVEL(NV)=D    THEN GOTO POP;                                   |
     378   3   2   |         LEVEL(NV)=LEVEL(NV)+1B;                                            |
     379   3   2   |         GOTO PUSH;                                                         |
     380   3   2   |         END RSLIST;                                                        |

                   |/*  'COMP' CALCULATES THE NUMBER OF WAYS FOR N VARIABLES                   */|
                   |/*  ,WITH SAME MAXIMUM LEVEL--D, TO SUM TO R--'ROW_SUM'                    */|
     381   2   2   |COMP:    PROC(N,R,D) RETURNS(FLOAT BIN(53));                                |
     382   3   2   |             DCL (N,R,D,M,I,MAX,R1,R2)FIXED BIN(15),                        |
                   |             (SUM,SIGN) FLOAT BIN(53);                                      |
     383   3   2   |             SIGN=-1E0B; SUM=0E0B;                                          |
     385   3   2   |             R1=R;  R2=D*N-R1;                                              |
     387   3   2   |             IF R1>R2 THEN R1=R2;                                           |
     388   3   2   |             MAX=R1/(D+1B);                                                 |
     389   3   2   |             DO M=0B TO MAX;                                                |
     390   3   3   |                I=R1-M*(D+1B);                                              |
     391   3   3   |                SIGN=-SIGN;                                                 |
     392   3   3   |             SUM=SUM+SIGN*COMB(N,M)*COMB(N+I-1B,I);                         |
     393   3   3   |             END;                                                           |
     394   3   2   |             RETURN(SUM);                                                   |

                   |/*  '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,N1,N2) FIXED BIN(15),                           |
                   |             (TOTAL,FACM) FLOAT BIN(53);                                    |
```

```
 STMT LEV NT

  397   4  2  |              IF M=0B THEN RETURN(1E0B);                                       |
  398   4  2  |              IF M=1B THEN RETURN(N);                                          |
  399   4  2  |              IF M=10B THEN RETURN( (N*(N-1B))/10B );                          |
  400   4  2  |              TOTAL,FACM=1E0B;                                                 |
  401   4  2  |              N1=N+1B-M; N2=M;                                                 |
  403   4  2  |              IF N-M >= M THEN GOTO CALC;                                      |
  404   4  2  |              N1=M+1B; N2=N-M;                                                 |
  406   4  2  |CALC:         DO TIMES=N TO N1 BY -1B;                                         |
  407   4  3  |                 TOTAL=TOTAL*TIMES;                                            |
  408   4  3  |              END;                                                             |
  409   4  2  |              DO TIMES=N2 TO 1B BY -1B;                                        |
  410   4  3  |                 FACM=FACM*TIMES;                                              |
  411   4  3  |              END;                                                             |
  412   4  2  |              RETURN(TOTAL/FACM);                                              |
  413   4  2  |              END COMB;                                                        |
  414   3  2  |END COMP;                                                                      |
  415   2  2  |NEXTO: END;                                                                    |

              |/*  PRINT FOR EACH CLASS EITHER THE SYMMETRIC SELECTOR(S) COVERING IT    |
              |    OR A MESSAGE INDICATING THAT NO SYMMETRIC VARIABLES WERE FOUND   */  |


  416   2  1  |PRINT:   IF ICL=0B THEN PUT SKIP FILE(SYSPRINT) EDIT ('CLASS','MIN',       |
              |                         'NUM. OF','CLASS FORMULI','COMPACT','O GROUP',    |
              |                         'SYM VARS','CLASS','RATIO','RAT(J','PER CLASS')   |
              |                         (X(85),A,X(6),A,X(6),A,SKIP,X(3),A,X(63),A,X(3),|
              |                         A,X(3),A,X(3),A,SKIP,X(85),A,X(5),A,X(4),A);     |
  417   2  1  |         IF DMAX#=0B THEN DO;PUT SKIP FILE(SYSPRINT) EDIT ('THERE ARE',|
              |                                     ' NO SYMMETRIC VARIABLES AND THERE',  |
              |                                     'FORE NO SYMMETRIC FORMULA FOR CLASS ',|
              |                                     ICL,DMAX#,ICL)(SKIP,A,A,A,F(3),X(20),  |
  419   2  2  |                                     F(3),X(5),F(3)); GOTO UNPRIME;   END;  |
  421   2  1  |         MINRATIO=1E0B;                                                   |
  422   2  1  |         DCL CHR(4) CHAR(1) ALIGNED INIT( '(','+','=',',' ),              |
              |              SIGN1 CHAR(2) ALIGNED,                                      |
              |             (SELCHR,COLS) FIXED BIN,SYMVAR(NV) BIT(1) ALIGNED,           |
              |             (#DCOMB,#COMB) FLOAT BIN(53);                                |
  423   2  1  |         SYMVAR='0'B; #DCOMB=1E0B;                                        |
  425   2  1  |         PUT FILE(SYSPRINT) EDIT                                          |
              |         ('F(',ICL,') := ')(SKIP,SKIP,A,F(2),A);                          |
  426   2  1  |         COUNT=10;                                                        |
  427   2  1  |         DO TOP=1B TO #DSUBTOP;                                           |
  428   2  2  |            SELCHR=1B;                                                    |
  429   2  2  |            IF TOP>1B THEN DO; PUT FILE(SYSPRINT) EDIT(' & ')(A);         |
  431   2  3  |                                  COUNT=COUNT+3;   END;                   |
  433   2  2  |            IF COUNT>70 THEN DO; PUT SKIP FILE(SYSPRINT) EDIT             |
  435   2  3  |                                      ('            ')(A);  COUNT=10;   END; |
  437   2  2  |            IV=DSUBTOP(TOP);                                              |
  438   2  2  |            DO WHILE(IV>0B);                                              |
```

```
PL/I OPTIMIZING COMPILER         MAINP:   PROC OPTIONS(MAIN) REORDER;

   STMT LEV NT

    439   2  3  |                SYMVAR(IV)='1'B;                                       |
    440   2  3  |                COLS=1B; IF  IV>1001B THEN COLS=10B;                   |
    442   2  3  |                IF COMPL(IV) THEN DO; PUT FILE(SYSPRINT) EDIT          |
                |                                   (CHR(SELCHR),'X',IV,'''')           |
                |                                   (A,A,F(COLS),A);                    |
    444   2  4  |                                   COUNT=COUNT+3+COLS;  END;           |
    446   2  3  |                             ELSE DO; PUT FILE(SYSPRINT) EDIT          |
                |                                   (CHR(SELCHR),'X',IV)                |
                |                                   (A,A,F(COLS));                      |
    448   2  4  |                                   COUNT=COUNT+2+COLS;  END;           |
    450   2  3  |                IF SELCHR=1B THEN SELCHR=10B;                          |
    451   2  3  |                IF COUNT>70 THEN DO; PUT SKIP FILE(SYSPRINT) EDIT      |
    453   2  4  |                                     ('         ')(A);  COUNT=10;  END; |
    455   2  3  |                IV=DMAXLINK(IV);                                       |
    456   2  3  |             END;                                                      |
    457   2  2  |          RS#=0E0B;                                                    |
    458   2  2  |             SELCHR=11B;                                               |
    459   2  2  |             DO IROW =ROW1 TO ROW2;                                    |
    460   2  3  |                IV=DSUBTOP(TOP); IRS=0B;                               |
    462   2  3  |                DO WHILE(IV>0B);                                       |
    463   2  4  |                   IRS=IRS+E(IROW,IV);                                 |
    464   2  4  |                   IV=DMAXLINK(IV);                                    |
    465   2  4  |                END;                                                   |
    466   2  3  |                RS#(IRS)=RS#(IRS)+1E0B;                                |
    467   2  3  |             END;                                                      |
    468   2  2  |             N=DSUBTOP#(TOP); D=DLIST(DSUBTOP(TOP));                   |
    470   2  2  |             #COMB=0E0B;                                               |
    471   2  2  |             DO IRS=0B TO D*N;                                         |
    472   2  3  |                IF RS#(IRS)=0E0B THEN GOTO NEXTRS;                     |
    473   2  3  |                COLS=1B; IF IRS>1001B THEN COLS=10B;                   |
    475   2  3  |                PUT FILE(SYSPRINT) EDIT                                |
                |                (CHR(SELCHR),IRS)(A,F(COLS));                          |
    476   2  3  |                COUNT=COUNT+1B+COLS;                                   |
    477   2  3  |                IF SELCHR=11B THEN SELCHR=100B;                        |
                |/*  CALCULATE CLASS COMPACT RATIO AND THE D_GROUP (OR SELECTOR)       |
                |    COMPACT RATIO                                                  */ |
    478   2  3  |                #COMP=COMP(N,IRS,D);                                   |
    479   2  3  |                #COMB=#COMB+#COMP;                                     |
    480   2  3  |                #REPETITIONS=RS#(IRS)/#COMP;                           |
    481   2  3  |                IF MOD(RS#(IRS),#COMP) ¬= 0 THEN                       |
                |                      #REPETITIONS=#REPETITIONS+1B;                    |
    482   2  3  |                RATIO=RS#(IRS)/(#REPETITIONS*#COMP);                   |
    483   2  3  |                IF RATIO<MINRATIO THEN MINRATIO=RATIO;                 |
    484   2  3  |NEXTRS:      END;                                                      |
    485   2  2  |             #DCOMB=#DCOMB*#COMB;                                      |
    486   2  2  |             PUT FILE(SYSPRINT) EDIT(')')(A);                          |
    487   2  2  |             COUNT=COUNT+1B;                                           |
    488   2  2  |          END;                                                         |
    489   2  1  |          DO IV=1B TO NV;                                              |
```

```
STMT LEV NT

 490   2  2 |             IF ¬SYMVAR(IV) THEN
            |                  #DCOMB=#DCOMB*(DLIST(IV)+1E08);
 491   2  2 |          END;
            |/*  PRINT RATIOS                                                  */ |
 492   2  1 |          PUT FILE(SYSPRINT) EDIT
            |          (#ROWS/#DCOMB,MINRATIO,DMAX#,ICL)(X(85-COUNT),F(6,3),
            |          X(4),F(6,3),X(7),F(3),X(6),F(2));


            |/*  RE_CONVERT THE VARIABLES IN THIS EVENT_CLASS                     |
            |    TO 'UNCOMPLEMENTED' FORM                                         |
            |    NOTE-- THIS IS REQUIRED WHEN FORMULAE FOR LATER CLASSES          |
            |    ARE VERIFIED AGAINST THIS EVENT_CLASS.                        */ |
 493   2  1 |UNPRIME: DO ID=1B TO NO;                                             |
 494   2  2 |             IV2,IV=DTOP(ID);                                         |
 495   2  2 |             DO WHILE(IV>0B);                                         |
 496   2  3 |                IF COMPL(IV) THEN                                     |
            |                              DO IROW=ROW1 TO ROW2;                  |
 497   2  4 |                                 E(IROW,IV)=DLIST(IV2)-E(IROW,IV);   |
 498   2  4 |                              END;                                   |
 499   2  3 |             IV=DLINK(IV);                                            |
 500   2  3 |             END;                                                     |
 501   2  2 |          END;                                                        |
 502   2  1 |          IF DMAX#¬=0B THEN RERUN='0'B;                               |


            |/*  GENERATE LIST OF SYMMETRIC VARIABLES COMMON TO ALL CLASSES WHICH |
            |    IS USED IN THE DATA REDUCTION TABLE                           */ |
 503   2  1 |          IF ¬DTRED THEN GOTO NEWCLASS;                               |
 504   2  1 |          IF ICL¬=0B THEN GOTO CHECK;                                 |
 505   2  1 |          SCOMPL=COMPL; SDMAXLINK=DMAXLINK;                           |
 507   2  1 |          SDSUBTOP=DSUBTOP;  SDSUBTOP#=DSUBTOP#;                      |
 509   2  1 |          S#DSUBTOP=#DSUBTOP;  GOTO NEWCLASS;                         |
 511   2  1 |CHECK:    IF S#DSUBTOP=0B THEN GOTO NEWCLASS;                         |
 512   2  1 |          DO I=1B TO S#DSUBTOP;                                       |
 513   2  2 |             DO J=1B TO #DSUBTOP;                                     |
 514   2  3 |                IF SDSUBTOP#(I)¬=DSUBTOP#(J) THEN GOTO A3;            |
 515   2  3 |                IV1=SDSUBTOP(I);  IV2=DSUBTOP(J);                     |
 517   2  3 |                DO WHILE (IV1>0B);                                    |
 518   2  4 |                   IF IV1¬=IV2 THEN GOTO A3;                          |
 519   2  4 |                   IF SCOMPL(IV1)¬=COMPL(IV2) THEN GOTO A3;           |
 520   2  4 |                   IV1=SDMAXLINK(IV1);  IV2=DMAXLINK(IV2);            |
 522   2  4 |                END;                                                  |
 523   2  3 |                GOTO A4;                                              |
 524   2  3 |A3:          END;                                                     |
 525   2  2 |             S#DSUBTOP=S#DSUBTOP-1B;                                  |
 526   2  2 |             SDSUBTOP(I)=0B;                                          |
 527   2  2 |A4:       END;                                                        |
```

```
PL/I OPTIMIZING COMPILER          MAINP:   PROC OPTIONS(MAIN) REORDER;

.   STMT LEV NT

    528   2  1  |NEWCLASS:  END;                                                          |
                |/*   IF THE SWITCH RERUN IS SPECIFIED AND NO SYMMETRIC VARIABLES WERE    |
                |     FOUND USING THE USER SPECIFIED SUBSET OF VARIABLES, THEN THE        |
                |     PROGRAM WILL AUTOMATICALLY RERUN ITSELF USING ALL THE VARIABLES     |
                |     AND ATTEMPT TO DETERMINE IF SYMMETRIC VARIABLES EXIST.          */  |

    529   2  0  | IF RERUN THEN DO; DLIST=DLIST+1B;                                       |
    531   2  1  |                            PUT SKIP FILE(SYSPRINT) EDIT('THIS JOB IS RE',  |
                |                            'RUN WITH THE SAME EVENTS AND AUTO SELECTION',  |
                |                            ' OF D GROUPS.')(A,A,A);                     |
    532   2  1  |                            GOTO LINKER;                                 |
    533   2  1  |                         END;                                            |
                |/*   IF THERE IS A SYMMETRY COVERING ALL  CLASSES (I.E. A SELECTOR       |
                |     WITH IDENTICAL REFEREE AND DIFFERENT REFERENCE FOR EACH CLASS),     |
                |     THE A NEW VARIABLE CAN BE DEFINED WHOSE VALUE IS THE SUM OF THE     |
                |     VALUES OF THE VARIABLES IN THE REFEREE.  THIS IS DATA REDUCTION.    |
                |     THE FOLLOWING ROUTINE TAKES SELECTORS OF THIS TYPE AND DISPLAYS THE|
                |     REDUCTION IN TABULAR FORM.                                       */|
    534   2  0  | IF ¬DTRED THEN GOTO NEWDATA;                                            |
    535   2  0  | IF SWDSUBTOP=0B THEN DO; PUT SKIP FILE(SYSPRINT) EDIT                  |
                |                              ('THERE ARE NO SETS OF SYMMETRIC VARI',    |
                |                               'ABLES COMMON TO ALL CLASSES.  THERE IS',  |
                |                               ' NO DATA REDUCTION AND THEREFORE',        |
                |                               'NO DATA REDUCTION TABLE.')(A,A,A,A);      |
    537   2  1  |                              GOTO NEWDATA;                               |
    538   2  1  |                          END;                                            |
    539   2  0  | PUT SKIP FILE(SYSPRINT) EDIT ('*****DATA REDUCTION TABLE*****',' ')     |
                |                                   (SKIP,SKIP,X(18),A,SKIP,SKIP,A);      |
    540   2  0  | J=0B;                                                                   |
    541   2  0  | DO I=1B TO NV;                                                          |
    542   2  1  |    IF SDSUBTOP(I)=0B THEN GOTO A5;                                      |
    543   2  1  |    J=J+1B;                                                              |
    544   2  1  |    IF (NV+J)<1010B THEN CL=1B;  ELSE CL=10B;                            |
    546   2  1  |    PUT SKIP FILE(SYSPRINT) EDIT ('VARIABLE X',NV+J,' IS THE SUM',       |
                |             ' OF VARIABLES')(A,F(CL),A,A);                              |
    547   2  1  |    IVI=SDSUBTOP(I);  SLC=1B;                                            |
    549   2  1  |    DO WHILE (IVI>0B);                                                   |
    550   2  2  |       DV=DLIST(IVI);                                                    |
    551   2  2  |       IF IVI<1010B THEN DL=1B;  ELSE DL=10B;                            |
    553   2  2  |       IF SCOMPL(IVI) THEN DO; PUT FILE(SYSPRINT) EDIT                   |
                |                                   (CHR2(SLC),'X',IVI,'''')(A,A,F(DL),A);|
    555   2  3  |                                   EV(*,J)=EV(*,J)+(DV-E(*,IVI));       |
    556   2  3  |                                END;                                     |
    557   2  2  |                         ELSE   DO; PUT FILE(SYSPRINT) EDIT              |
                |                                   (CHR2(SLC),'X',IVI)(A,A,F(DL));       |
    559   2  3  |                                   EV(*,J)=EV(*,J)+E(*,IVI);             |
```

```
PL/I OPTIMIZING COMPILER          MAINP:    PROC OPTIONS(MAIN) REORDER;

    STMT LEV NT

     560   2  3  |                          END;
     561   2  2  |         IF SLC=1B THEN SLC=10B;
     562   2  2  |         IVL=SOMAXLINK(IVL);
     563   2  2  |      END;
     564   2  1  |A5:END;
     565   2  0  |    PUT SKIP FILE(SYSPRINT) EDIT (' ')(SKIP,SKIP,X(14),A);
     566   2  0  |    DO I=1B TO NV;
     567   2  1  |      IF I>1001B THEN CL=10B;  ELSE CL=1B;
     569   2  1  |     PUT FILE(SYSPRINT) EDIT ('X',I)(X(100B-CL),A,F(CL));
     570   2  1  |    END;
     571   2  0  |    PUT FILE(SYSPRINT) EDIT ('  |')(A);
     572   2  0  |    DO I=NV+1B TO NV+J;
     573   2  1  |       IF I>1001B THEN CL=10B;  ELSE CL=1B;
     575   2  1  |      PUT FILE(SYSPRINT) EDIT ('X',I)(X(100B-CL),A,F(CL));
     576   2  1  |    END;
     577   2  0  |    PUT FILE(SYSPRINT) EDIT
                 |    (('CLASS F(',ICL,')',
                 |       ('EVENT NO.',IROW-ROWCL(ICL),'=',E(IROW,*),'  |',
                 |       (EV(IROW,ICOL2) DO ICOL2=1B TO J) DO IROW=ROWCL(ICL)+1B
                 |        TO ROWCL(ICL)+NE(ICL)) DO ICL=0B TO MAXCL))
                 |    ((NCL)(SKIP,A,F(2),A,(NE(ICL))(SKIP,A,F(5),A,(NV)(F(5)),A,
                 |     (J)(F(5))))));

     578   2  0  |NEWDATA:  END;
     579   1  0  |             GOTO NEXT;
     580   1  0  |EXIT: END MAINP;
```