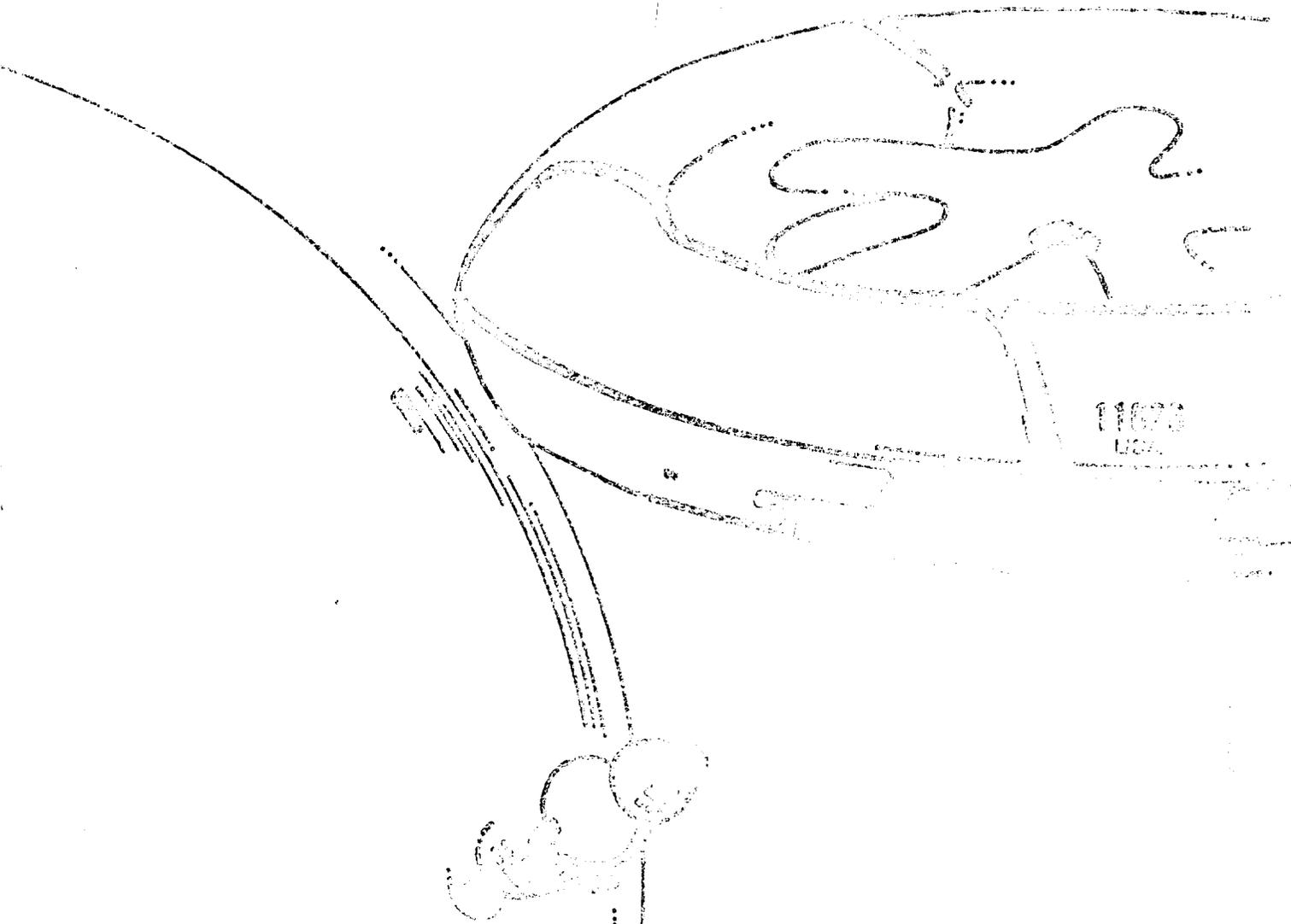


MATHEMATICAL MODEL OF SUPPLY  
SUPPORT FOR SPACE OPERATIONS



11070  
127

X71-10222

(ACCESSION NUMBER)

N75-71353

(NASA-CR-114835) MATHEMATICAL MODEL OF  
SUPPLY SUPPORT FOR SPACE OPERATIONS: USER'S  
MANUAL AND OPERATING INSTRUCTIONS FOR  
COMPUTER PROGRAM (Combinatorics, Inc.) 90 p

Unclas  
00/98 48484

----- GOVERNMENT AGENCIES ONLY



CR-114825

MATHEMATICAL MODEL OF SUPPLY  
SUPPORT FOR SPACE OPERATIONS

USER'S MANUAL  
and  
OPERATING INSTRUCTIONS  
for  
COMPUTER PROGRAM

VAS-11329

Prepared for:

National Aeronautics and Space Administration  
Flight Operations Support Section  
Manned Spacecraft Center  
Houston, Texas

by

COMTECHNICS, INC.  
3047 Leavenworth  
Oakland, California 94612

## TABLE OF CONTENTS

### USER'S MANUAL FOR THE MMSSSO PROGRAM

Introduction . . . . .	1
Mathematical Development . . . . .	11
Computer Program . . . . .	20
Glossary . . . . .	56
Appendix I . . . . .	60
Appendix II . . . . .	67
References . . . . .	75

### OPERATING INSTRUCTIONS FOR MMSSSO COMPUTER PROGRAM

Input . . . . .	1
Output . . . . .	6
Appendix . . . . .	8
Example 1 (INPUT-DATA)	
Example 2 (Normal Output)	
Example 3 (Normal Output after Switches)	
Example 4 (Error Checking Output)	

## USER'S MANUAL FOR THE MMSSSO PROGRAM

### Introduction

This program is concerned with supply support for space bases. A plan of operations, reflecting scientific as well as regular living activities at the space base, is assumed to exist. From this plan and the hardware technology to be used at the base a schedule of requirements is generated. These requirements are in the form of standardized deliverable modules. Associated with each module is an earliest and latest time determined by the schedule of requirements and the allowable holding time for the contents of the module. Holding time is, of course, a design parameter controlled by the storage capacity built into the space base. In addition to the time interval for each module it is assumed its loading characteristics such as weight, diameter and length are known. The problem is to fulfill the schedule of requirements via a series of trips by specifying the cargoes for each load in such a manner that a time interval exists for scheduling the trip to deliver the load. The objective, of course, is to construct the loads in such a manner as to minimize the number of trips required to satisfy the schedule of requirements. In general, the derivation of demands for a space base is a highly complex process. The technique of activity analysis could be extremely useful in studying the simultaneous interaction of demands and activity levels as reflected in various plans of operations. In such an analysis, the coefficients could be determined by the

types of technology chosen to implement in the space base,

In attempting to formulate the space base supply problem as a mathematical problem, it would be natural to think first in terms of the classical assignment problem. The classical assignment problem provides a model for assigning a set of objects to a set of locations with a fixed cost for each possible assignment.

### Model I Classical Assignment Problem

Let,

$$I_{ij} = \begin{cases} 1 & \text{if module } i \text{ is assigned to load } j \\ 0 & \text{otherwise} \end{cases}$$

$$q_{ij} = \text{cost of assigning module } i \text{ to load } j$$

$$b_j = \text{bound on number of modules assignable to load } j$$

then subject to

$$\sum_j I_{ij} = 1 \quad \text{for each } i \text{ (each module must be assigned)}$$

$$\sum_i I_{ij} \leq b_j \quad \text{for each } j \text{ (not more than capacity of load)}$$

$$\min \sum_{i,j} q_{ij} I_{ij}$$

A slight extension would be to the Capacitated Assignment Problem. This would entail adjoining capacity constraints of the form

$$I_{ij} = 0 \quad i, j \in S$$

where  $S$  defines the pairs  $(i, j)$  that are not allowable. The greatest advantage of this model is that the discrete 0 or 1 nature of the variables  $I_{ij}$  need not be imposed directly. The  $I_{ij}$  variables can be treated as positive continuous variables

and the nature of the constraints will insure that the variables are naturally discrete 0 or 1 variables at the optimal solution. There are very efficient algorithms for solving this model (see reference 1 for instance). The obvious problem with this model is that the cost associated with assignment of a module to a load is completely independent of which other modules are also assigned to the load. This model could not prevent modules with non-overlapping time intervals or other packing incompatibilities from being scheduled together. It would, in fact, be useful only if we had substantially different vehicles and the carrying costs of the modules by different vehicles was of principal interest. This model does not reflect relations between modules at all only the relation between module and vehicle.

Abandoning the classical assignment problem, a more general linear integer programming problem could be set up along the lines of the well-known "Set Covering" or "Warehouse Location" models. This approach would require the generation of feasible configurations. A feasible configuration for a load would be a specification of a pre-selected time interval and a specification of number and type of modules which would make a compatible load. Having generated the feasible configurations. A list of feasible configurations containing selected duplications is assembled. Given this list of configurations, the general linear integer programming model can be setup.

Model II General Integer Programming Problem

Let,

$$I_{ij} = \begin{cases} 1 & \text{if module } i \text{ is assigned to configuration } j \\ 0 & \text{otherwise} \end{cases}$$

$$o_j = \begin{cases} 1 & \text{if configuration } j \text{ is open} \\ 0 & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1 & \text{if module } i \text{ can be assigned to configuration } j \\ 0 & \text{otherwise} \end{cases}$$

$k_j$  = capital cost of configuration  $j$

$q_{ij}$  = cost of assigning module  $i$  to configuration  $j$

then subject to

$$\sum_j a_{ij} I_{ij} = 1 \quad \text{for each } i \text{ (each module must be assigned)}$$

$$\sum_i I_{ij} \leq o_j \quad \text{for each } j \text{ (must be assigned to open configuration)}$$

$$\min \sum_j k_j o_j + \sum_{i,j} q_{ij} I_{ij}$$

Using Bender's decomposition which would yield capacitated assignment problems for the "sub-problems", it might be possible to solve this model despite the large number of configurations that would be required. The large integer programming problem that would constitute the "master" program would still present a very formidable computational task. The assembling of the possible lists of feasible configurations would also be somewhat arbitrary and might drastically effect the solutions. Assessing the costs for a configuration and the assignment of a module to a configuration in terms of how effectively they utilize the capacity of the vehicle might also be quite difficult and expensive. On the whole, it would seem

The "0" priority class is reserved for locking modules on vehicles when they must be assigned to a unique vehicle as illustrated by the fifth "X-1" module. This designation is also used to achieve unequal loading by locking fictitious modules with "weights" equal to the desired differentials on the lower weight capacity vehicles while leaving the time intervals wider than the planning horizon. See page 16 of the User's Manual for an illustration and more complete discussion. Conversely, vehicles can be locked to time intervals by specifying time intervals and leaving the physical descriptors as zero.

The input is immediately output for verification. The namelist card is output by the namelist and the parameters are appropriately labeled. The module descriptors are output under the labeling given by format statement 12 in the routine "MAIN". The labels are:

- No.        - number of module
- Module     - alpha-numeric descriptor
- Weight     - weight of module in thousands of pounds
- Diameter   - diameter of module in feet
- Length     - length of module in feet
- TE         - earliest time for vehicle
- TF         - final time for vehicle
- LE         - first vehicle possible for the module
- LF         - final vehicle possible for the module
- NC         - number of priority class
- IV         - initial assignment of module to a vehicle

OUTPUT

The normal output from the program is generated in the subroutine "PRINT" and is illustrated by Example 2. This output gives first the four parameters:

- IX1:** The value IX1=0 indicates the modules are free to move over all vehicles, while the value IX1=1 indicates the modules are restricted to higher order interchanges between vehicles. This flag indicates the current mode of search procedure.
- GAP:** The length of the time interval specified for the vehicles in the current run.
- F:** The current emphasis between the time intervals and the weighted combination of physical descriptors. As an illustration if FAC=3 this value will move through the values  $F=1, F=2/3, F=1/3, F=0$  which give the parametric weight assigned to the time interval as opposed to the physical characteristics.
- G:** This gives the current value of the interference function being minimized. It is negative when  $FAC > 0$  and there are time incompatibilities in the initial assignments.

These parameters are followed by a summary of the present assignments giving the vehicle number, the earliest time, the latest time, the total weight, the total diameter, and the total length followed by the list of modules currently assigned to the vehicle. These quantities are output under the labeling given by format statement 3 in the subroutine "PRINT". The labels are:

- LOAD** - the number of the load or vehicle
- TMIN** - the earliest time the vehicle can be dispatched as determined by the intersection of the time intervals of the individual modules currently assigned to the vehicle.

- TMAX - the latest time the vehical can be dispatched as determined by the intersection of the time intervals of the individual modules currently assigned to the vehicle
- WEIGHT - the sum of the weights of the individual modules assigned to the vehicle
- DIAMETER - the sum of the diameters of the individual modules assigned to the vehicle
- LENGTH - the sum of the lengths of the individual modules assigned to the vehicle
- MØDULES - a list of the modules currently assigned to the vehicle

In terms of the example the output would appear as follows:

LOAD	TMIN	TMAX	WEIGHT	DIAMETER	LENGTH	MØDULES
1	0.0	12.0	35.036	10.0	10.0	5
2	0.0	12.0	32.06	11.0	34.0	1 3
3	13.0	24.0	32.06	11.0	34.0	2 4

There are two output statements in the program calling "PRINT". These "CALL" statements are in the subroutine "SETUP". They cause a summary output whenever a priority class has been completely scanned for reducing interchanges and such reducing interchanges have been found decreasing the value of the interference function. Compare Example 2 and Example 3 for an illustration of this. See the flow diagrams on pages 30 and 31 of the User's Manual for precise information as to when this output is generated.

APPENDIXOutput from "PRINTE"

The information generated from this subroutine is not required for normal operation of the program. It can only be understood after study of the User's Manual. It is only employed for detailed monitoring of the internal behavior of the program. The output from this subroutine is illustrated in Example 4. The input parameter IPRT=1 is used to turn on this intermediate print out. The first line of output gives with labels:

G:           current value of interference functional  
 GH:          amount of gain or loss in current trial sequence of interchanges

The second line of output is the label "assignments". This is immediately followed by the current maps printed with FORMAT (1H□39I3):

MAPR(IP):    module number to be assigned  
 MAPC(IP):    vehicle number to which the corresponding module in MAPR(IP) can be assigned  
 MAPP(IP):    list of modules in priority class currently being scanned

This is followed by the labeled variables:

M2:          number of moveable modules in priority class being scanned  
 IX1:         at value IX1=0 movement of the modules over the entire range of vehicles is being considered while at value IX1=1 movement is restricted to interchanges between pairs of vehicles

$M = \{1, \dots, m\}$  of modules into the set  $V = \{1, \dots, v\}$  of vehicles.

Such a map would take the form

$$p = \begin{pmatrix} 1 & 2 & \dots & m \\ 3 & 4 & \dots & 17 \end{pmatrix}$$

where the top line lists the module and the element below on the second line gives the number of the vehicle to which the module is assigned. [See references 2 and 3 for a completely general scheme for enumerating such finite maps.] The space supply support problem can be expressed simply in terms of such mappings

#### Model V Space Supply Support by Mappings

Let,  $S$  be the set of all mappings of the set  $M = \{1, \dots, m\}$  into the set  $V = \{1, \dots, v\}$ . Thus a particular  $\rho \in S$  can be represented in the form

$$\rho = (s_1, s_2, \dots, s_m)$$

where  $s_j$  indicates the vehicle to which module  $j$  is assigned.

Also let,

$$q_{ik} = \text{interference of modules } i \text{ and } k \text{ where } q_{ik} = q_{ki} \\ \text{and } q_{ii} = 0$$

$$S_t = \{i \mid \rho(i) = t\}$$

Now consider the functionals

$$\varphi_{it} : S \rightarrow R'$$

defined as

$$\varphi_{it}(\rho) = \sum_{r \in S_t} q_{ir}$$

and the composite functional

$$\varphi : S \rightarrow R'$$

defined as

$$\varphi(\rho) = 1/2 \sum_{t \in V} \sum_{i \in S_t} \varphi_{it}(\rho)$$

The problem reduces to:

$$\min_{\rho \in S} \varphi(\rho)$$

(Note all the constraints are absorbed into the generation of the maps.) This space of possible mapping of modules into vehicles although finite and enumerable in theory is extremely large for it contains  $v^m$  maps. In practice even with moderate numbers of modules and vehicles and given the most advanced computing equipment, it is not possible to completely explicitly enumerate all possible maps. The alternatives are to implicitly completely enumerate and to abandon the complete enumeration in favor of a restricted enumeration of a limited subset of maps. The first alternative, a complete implicit enumeration would rely on implicitly enumerating large subsets of maps by demonstrating from logically derived bounds that such a set could not contain a better map. Such bounds for the space supply support program will be derived in the next section. The enumeration scheme employed in the present computer program is further designed to keep open the option of whether to go to the complete implicit enumeration or whether to terminate after searching a restricted subset of maps. The generation of the maps in this computer program differs radically from the generation scheme proposed in reference 2 and 3. It is designed to utilize the particular nature of the truncating

bounds employed and consists of generating excursions from a given map by exchanging modules between vehicles. The present program might well be strengthened by employing the global techniques developed in detail in references 2 and 3 to obtain the initial map from which to undertake the excursions. These techniques might eliminate a large number of local exchanges and greatly speed up the computation.

### Mathematical Development

As elaborated in the previous section in Model V, the mathematical objective is to minimize the total interference function over the space of all possible assignments of modules to vehicles or

$$\min_{\rho \in S} \varphi(\rho) .$$

Given any mapping

$$\rho = (s_1, \dots, s_m)$$

any other mapping  $\rho'$  can be reached from  $\rho$  by a sequence of single switches. This simply means that the total number of reassignments can be achieved by choosing a single assignment

$$i \rightarrow \mu \quad \text{and switching it to} \quad i \rightarrow \omega$$

then choosing a second assignment

$$j \rightarrow s \quad \text{and switching it to} \quad j \rightarrow t$$

and so on until all the required reassignments of modules to vehicles has been accomplished. Our first objective is to derive the effect of such a single switch on the total interference function  $\varphi$ .

Transition Formula

Suppose the assignment of module  $j$  is switched from vehicle  $s$  to vehicle  $t$ , e.g., from  $j-s$  to  $j-t$ . Consider first the effect on the

$$\varphi_{iu} = \sum_{r \in S_u} q_{ir}$$

where  $S_u = \{r \mid \rho(r) = u\}$ .

The effects can be summarized as follows:

- 1)  $S'_u = S_u$  for  $u \neq s, t$ ;  $S'_s = (S_s - j)$ ;  $S'_t = (S_t \cup j)$
  - 2)  $\varphi'_{iu} = \varphi_{iu}$  for  $u \neq s$  or  $t$  (no effect)
  - 3)  $\varphi'_{is} = \varphi_{is} - q_{ij}$
  - 4)  $\varphi'_{it} = \varphi_{it} + q_{ij}$
  - 5)  $\varphi'_{js} = \varphi_{js}$
  - 6)  $\varphi'_{jt} = \varphi_{jt}$
- } for  $i \neq j$
- } Since  $q_{jj} = 0$

Now consider the effect on the total interference functional

$$\varphi = 1/2 \sum_{u \in V} \sum_{i \in S_u} \varphi_{iu}$$

The first step is to break this functional after the transition down into its relevant components

$$\begin{aligned} \varphi' &= 1/2 \sum_{u \in V} \sum_{i \in S'_u} \varphi'_{iu} \\ &= 1/2 \left\{ \sum_{\substack{u \in V \\ u \neq s, t}} \sum_{i \in S'_u} \varphi'_{iu} + \sum_{i \in S'_s} \varphi'_{is} + \sum_{i \in S'_t} \varphi'_{it} \right\} \end{aligned}$$

Since  $S'_u = S_u$  and  $\varphi'_{iu} = \varphi_{iu}$  for  $u \neq s, t$  the first component is invariant,

$$\sum_{\substack{u \in V \\ u \neq s, t}} \sum_{i \in S'_u} \varphi'_{iu} = \sum_{\substack{u \in V \\ u \neq s, t}} \sum_{i \in S_u} \varphi_{iu}$$

The second component can be reduced to the original terms as follows:

$$\sum_{i \in S'_s} \varphi'_{is} = \sum_{i \in S'_s} \varphi_{is} - \varphi_{js} - \sum_{i \in S'_s} q_{ij}$$

$$\sum_{i \in S'_s} q_{ij} = \sum_{i \in S_s} q_{ji} = \varphi_{js} \text{ since } q_{ij} = q_{ji} \text{ and } q_{jj} = 0$$

hence,

$$\sum_{i \in S'_s} \varphi'_{is} = \sum_{i \in S_s} \varphi_{is} - 2 \varphi_{js}$$

The third component can be reduced to the original terms as follows:

$$\sum_{i \in S'_t} \varphi'_{it} = \sum_{i \in S_t} \varphi_{it} + \varphi_{jt} + \sum_{i \in S'_t} q_{ij}$$

$$\sum_{i \in S'_t} q_{ij} = \sum_{i \in S_t} q_{ji} = \varphi_{jt} \text{ since } q_{ij} = q_{ji} \text{ and } q_{jj} = 0$$

hence,

$$\sum_{i \in S'_t} \varphi'_{it} = \sum_{i \in S_t} \varphi_{it} + 2 \varphi_{jt}$$

Reconstituting the components

$$\varphi' = \varphi + (\varphi_{jt} - \varphi_{js})$$

This is the crucial formula employed in the Space Supply Support Program. The  $\varphi_{iu}$  are computed initially and then the critical differences  $(\varphi_{it} - \varphi_{is})$  are found and updated from the transition formulas 2) - 6) as a sequence of trial switches are tried. These differences  $(\varphi_{it} - \varphi_{is})$  immediately implicitly enumerate all single switches when they are non-negative and guarantee a gain when negative. However, note if we were to switch,

from  $j \rightarrow s$  to  $j \rightarrow t$

and from  $k \rightarrow t$  to  $k \rightarrow s$

a double exchange between vehicles,

$$\varphi'' = \varphi + (\varphi_{jt} - \varphi_{js}) + (\varphi_{ks} - \varphi_{kt}) - 2 \varphi_{jk}$$

and even when the first order switches cannot reduce the interference functional a gain can be achieved with double and higher order switches. It should be stressed that whenever a switch or a sequence of trial switches achieves a reduction in the total interference functional  $\varphi(\rho)$ , the current map  $\rho$  is updated to  $\rho'$ . The process then starts over considering first order then second order and on up switches to a pre-set level.

Another significant aspect of the particular form of the interference functional employed

$$\varphi_{ij} = p_1 w_i w_j + p_2 d_i d_j + p_3 l_i l_j$$

is revealed when we drop back to the variable formulation of the problem given in Model IV and consider the smoothed continuous version of the problem where the variables  $I_{ij}$  in that

formulation are taken as continuous variables. In this continuous version of the problem we wish to minimize

$$\varphi = \sum_{i>j} q_{ij} \left( \sum_p I_{ip} I_{jp} \right)$$

subject to the side constraints

$$\sum_p I_{ip} = 1 .$$

Solving for the variables

$$I_{i1} = \left( 1 - \sum_{p>1} I_{ip} \right)$$

and substituting into the extremal function, we wish to minimize

$$\varphi = \sum_{i>j} q_{ij} \left[ \left( 1 - \sum_{p>1} I_{ip} \right) \cdot \left( 1 - \sum_{p>1} I_{jp} \right) + \sum_{p>1} I_{ip} \cdot I_{jp} \right]$$

The minimum is obtained by equating the partial derivatives to zero.

$$\begin{aligned} \frac{\partial \varphi}{\partial I_{ts}} &= \sum_{k \neq t} q_{tk} \left( -1 + \sum_{p>1} I_{kp} \right) + I_{ks} \\ &= \sum_{k \neq t} q_{tk} \left( -I_{k1} + I_{ks} \right) = 0 \end{aligned}$$

for  $t=1, \dots, n$  and  $s=2, \dots, v$ .

Hence, at the stationary point

$$\sum_{k \neq t} q_{tk} \cdot I_{k1} = \sum_{k \neq t} q_{tk} \cdot I_{ks} .$$

In the case where  $q_{tk} = w_t \cdot w_k$  and  $q_{tt} = 0$

$$\sum_{k \neq t} w_t \cdot w_k \cdot I_{k1} = \sum_{k \neq t} w_t w_k \cdot I_{ks}$$

and factoring out the  $w_t$  yields the equal loading phenomena on each vehicle,

$$\sum_k w_k I_{k1} = \sum_k w_k \cdot I_{ks} .$$

This continuous result does not answer directly what happens in our discrete case. Remarkly however, this effect does persist in the discrete case and extensive computational experience verifies that apart from the time interval anomalies we achieve very closely this equal loading phenomena. It would seem possible to demonstrate this result directly in the discrete case and in fact to bound the deviations in terms of first order switches, second order switches and so on up. This would be a highly desirable result in terms of determining the level of switches to be explored in the program, however no precise statements in this direction can be made at this time. Note that the vehicle loading can easily be unbalanced as desired by "locking" fictitious modules on specific vehicles. Specifically, a module with weight equal to the difference between the capacity of the vehicle and the maximum capacity of any vehical in the program is added to each vehicle and prevented from moving by the partioning mechanism to be described next. Of course, the time intervals for the fictitious modules must encompass the planning horizon so as not to create any time interval interference. This unbalancing perhaps can best be understood in terms of the following simple illustration.

Illustration of Unbalancing

	Vehicle 1	Vehicle 2	Vehicle 3
Real Capacity	20	30	40
Fictitious Module	20	10	0
Balanced Weight	40-S	40-S	40-S
True Weight	(20-S)	(30-S)	(40-S)

The quantity  $S = \frac{c-w}{v}$  would be very close, except for discreteness factors, to the difference between total capacity  $c$  and total weight to be assigned  $w$  divided equally between the  $v$  vehicles.

In order to extend the capacity of the program to very large numbers of modules and vehicles without incurring the consequent exponential increase in computational burden, two distinct optional partitioning mechanisms are incorporated. The first procedure partitions the modules into priority classes. Let  $n$  be the total number of modules to be assigned and  $n_1, n_2, \dots, n_c$  be the number in each of  $c$  priority classes. In considering single switches there would be  $n \cdot v$  such possible single switches where  $v$  is the number of vehicles. In considering each priority class singly and sequentially we would consider  $n_1 \cdot v + n_2 \cdot v + \dots + n_c \cdot v = n \cdot v$  since of course  $n_1 + \dots + n_c = n$ . Hence, there is no appreciable gain except for storage in the computer and some reduction in updating costs on the differences  $(\varphi_{js} - \varphi_{jt})$ . Also there is no loss in program effectiveness because each module is ultimately allowed to move. However, when we consider pairwise and higher switches the situation is quite different. Roughly we would be comparing  $\binom{n}{2}$  to  $\binom{n_1}{2} + \dots + \binom{n_c}{2}$ . As an example with  $n = 8$ , we would be comparing

$$\binom{8}{2} = \frac{8 \cdot 7}{2} = 28 \text{ to say } \binom{4}{2} + \binom{4}{2} = 6 + 6 = 12.$$

(Roughly because there is a slight correction because we would not consider switches on the same vehicle.) With large numbers of modules the work is enormously reduced. There is, of course,

some reduction in effectiveness. However, if modules of approximately the same weight are grouped together, the loss of effectiveness should not be very significant. In general about 30 modules to a priority class would seem most appropriate. Of course, modules with large time interval overlaps are also good candidates for grouping in the same category.

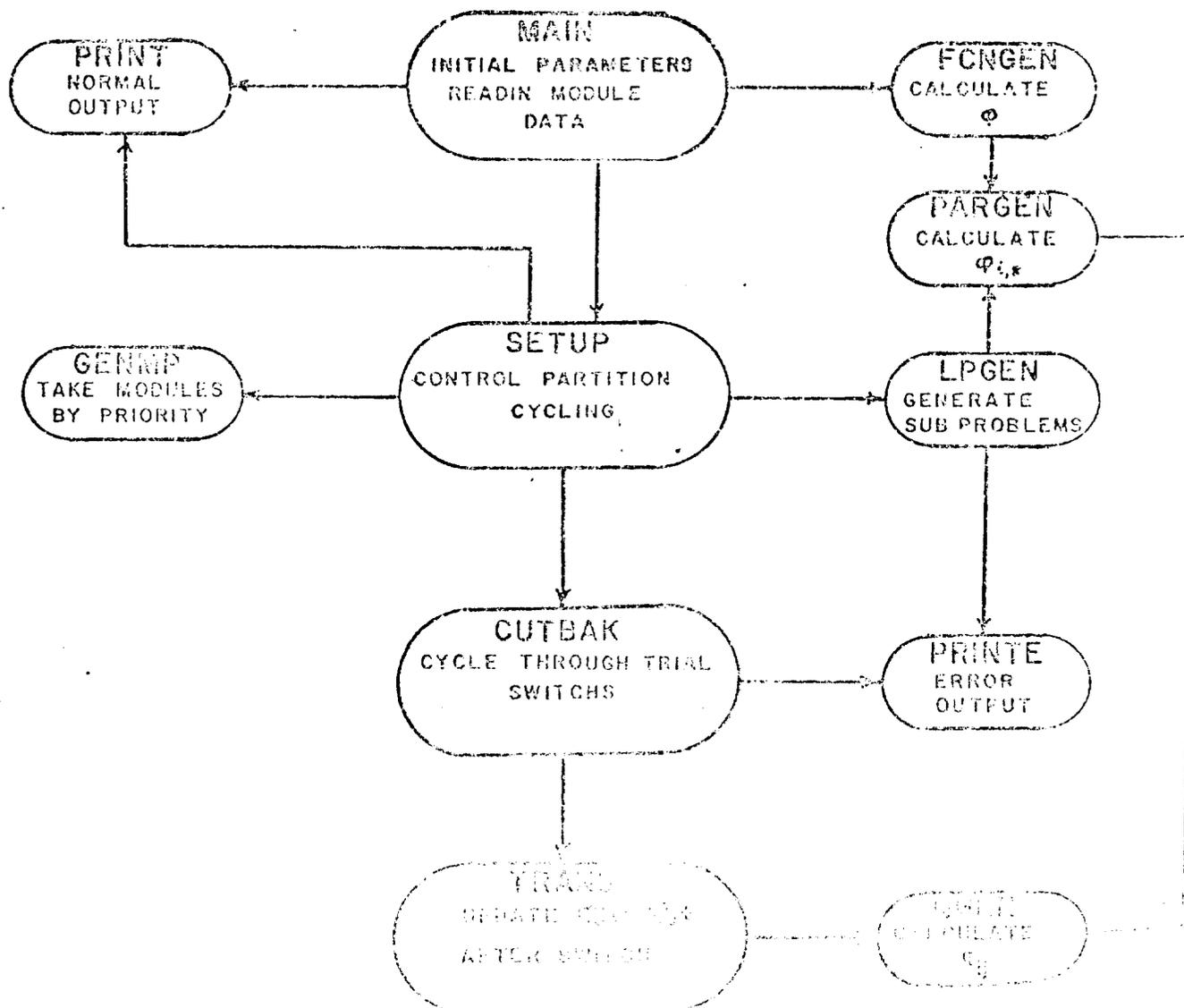
The second optimal partitioning scheme permits the restriction of trial interchanges to interchanges between pairs of vehicles. There are  $\frac{v \cdot (v-1)}{2}$  such possible distinct pairs of vehicles. Each pair is considered in turn sequentially and the pair is balanced. The transitivity effect allows a module to migrate sequentially to any permissible vehicle. No vehicle can remain grossly unbalanced for it would ultimately be forced to exchange with another vehicle to achieve the minimization of the interference functional.

This second partitioning scheme and interchanges between pairwise vehicles is accomplished within priority classes. When all the modules are lumped into the same priority class and the pairwise vehicle restriction is not imposed and the level of permissible interchanges raised sufficiently high a complete enumeration is attained. In practice, this is generally not possible with problems of real interest. However, empirically evidence shows that there is very little degradation of the final solution with the use of the partitioning schemes and restricting the trial level of exchanges to pairwise between modules. A more technical elaboration of the

the partitioning schemes is contained in the next section in conjunction with the detailed analysis of the computer program itself.

THE COMPUTER PROGRAM

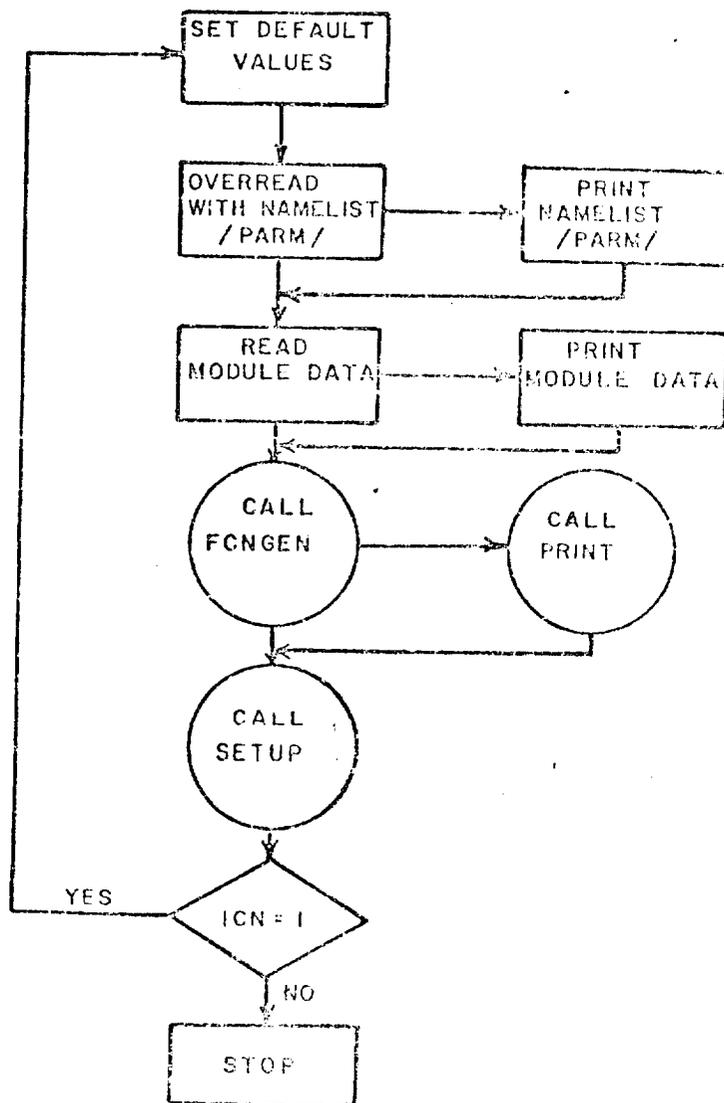
The computer program consists of eleven subroutines. Each of these subroutines will be described separately in terms of their relation to each other and in relation to the underlying mathematics. A general glossary is supplied defining all the vectors and variables in the common statement. Variables specific to each subroutine will be defined in the discussion of the relevant subroutine. The interactions between the eleven subroutines are consisely summarized in the following:

GENERAL FLOW DIAGRAM

ROUTINE MAIN

The purpose of this routine is to initialize the program parameters and to read in the data required for each module. The sequence of the program parameters is given in the "Operating Instructions for MMSSSO Computer Code" and the general flow is as follows:

ROUTINE MAIN FLOW DIAGRAM



The namelist statement is:

```
NAMELIST/PARM/NPL,NST,LV1,LV2,GAP,FAC,P1,P2,P3,IPRT,ICN,N
```

The format statements employed are:

```
11 FØRMAT(A4,5F6.0,4I2)
```

```
12 FØRMAT('NO. MØDULE WEIGHT DIAMETER LENGTH TE TF ILE LF NC IY')
```

```
13 FØRMAT(1H I3,2X,A4,3F10.2,2F8.1,4I4)
```

The 11 FØRMAT is used to readin the module data.

The 12 FØRMAT is used to label the module data output.

The 13 FØRMAT is used to print the module data.

The variables not in common are:

ICN - The value ICN=0 causes the program to stop after completing a run. The value ICN=1 causes the program to start over with a new set of data.

IP - An index variable to readin the successive modules.

X - A dummy variable to readin and print out the alphanumeric descriptor of a module.

N - The value N=0 allows the program to run stacked cases using payload module data carried over from the previous case. The value N=1 allows payload data definitions to be input.

Subroutine FCNGEN

The purpose of this subroutine is to make an initial computation of  $\varphi(\rho) = 1/2 \sum_{t \in V} \sum_{i \in S_t} \varphi_{it}(\rho)$  where

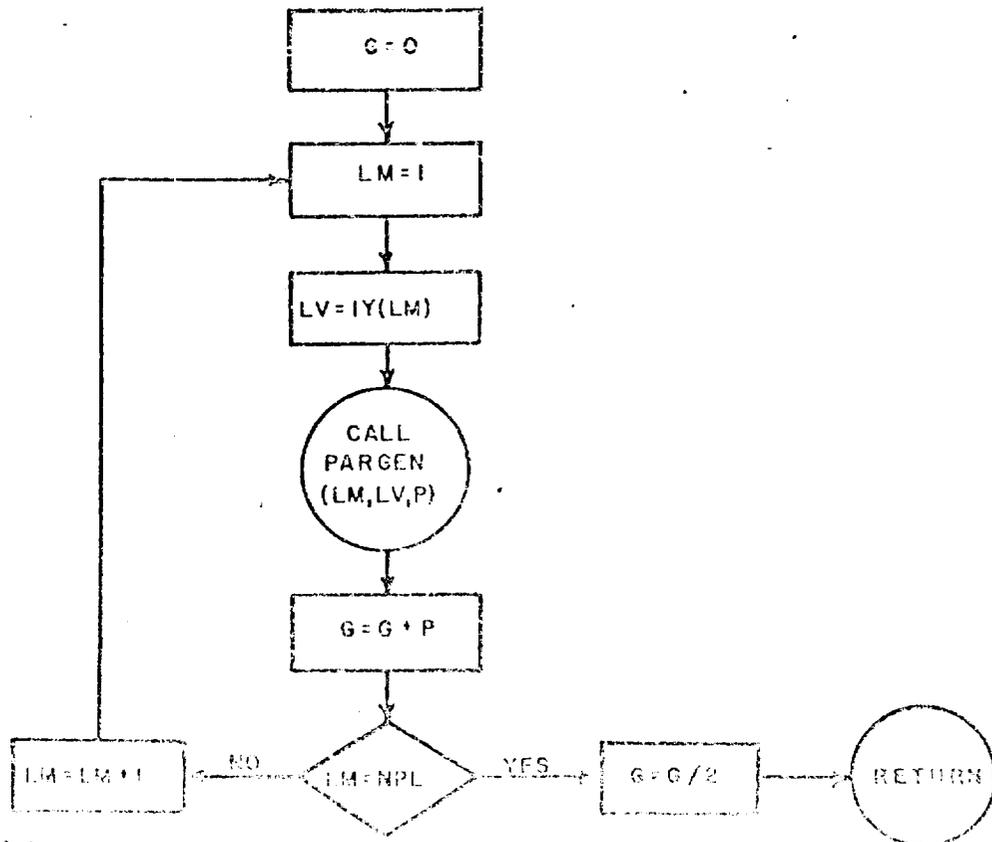
$$\varphi_{it}(\rho) = \sum_{r \in S_t} q_{ir}$$

and  $S_t = \{r \mid \rho(r) = t\}$

and  $q_{ir}$  = interference of module  $i$  and  $r$  where

$$q_{ir} = q_{ri}, \quad q_{ii} = 0.$$

After the initial computation  $\varphi(\rho)$  is simply updated as switches are made.

SUBROUTINE FCNGEN FLOW DIAGRAM

The variables not in common are:

- 1 - Index the modules
- 2 - Gives the vehicle to which LM is currently assigned.
- 3 - Returns the value of  $q_{LM, LV}$

Subroutine PRINT

The purpose of this subroutine is to print a summary of the present assignments giving the vehicle number, the earliest time, the latest time, the total weight, the total diameter, and the total length followed by the list of modules currently assigned to the vehicle. The meaning of the parameters also printed out is given in the "Operating Instructions for MMSSSO Computer Code" and the general Glossary.

The format statements employed are:

```
1 FØRMA T(5H11x1=I2,5H GAP=F2.0,3H F=F6.4/3H G=E16.8)
3 FØRMA T('LOAD TMIN TMAX WEIGHT DIAMETER LENGTH MØDULES')
4 FØRMA T(1H I3,F7.6,F6.1,3F8.3,2H 15I15/(1H 26X,20I5))
```

The 1 FØRMA T is used to print the parameters.

The 3 FØRMA T is used to print the labels.

The 4 FØRMA T is used to print the vehicle summary.

The variables not in common are:

IQ - Indexes the vehicles.

MC - Counts the modules on a particular vehicle.

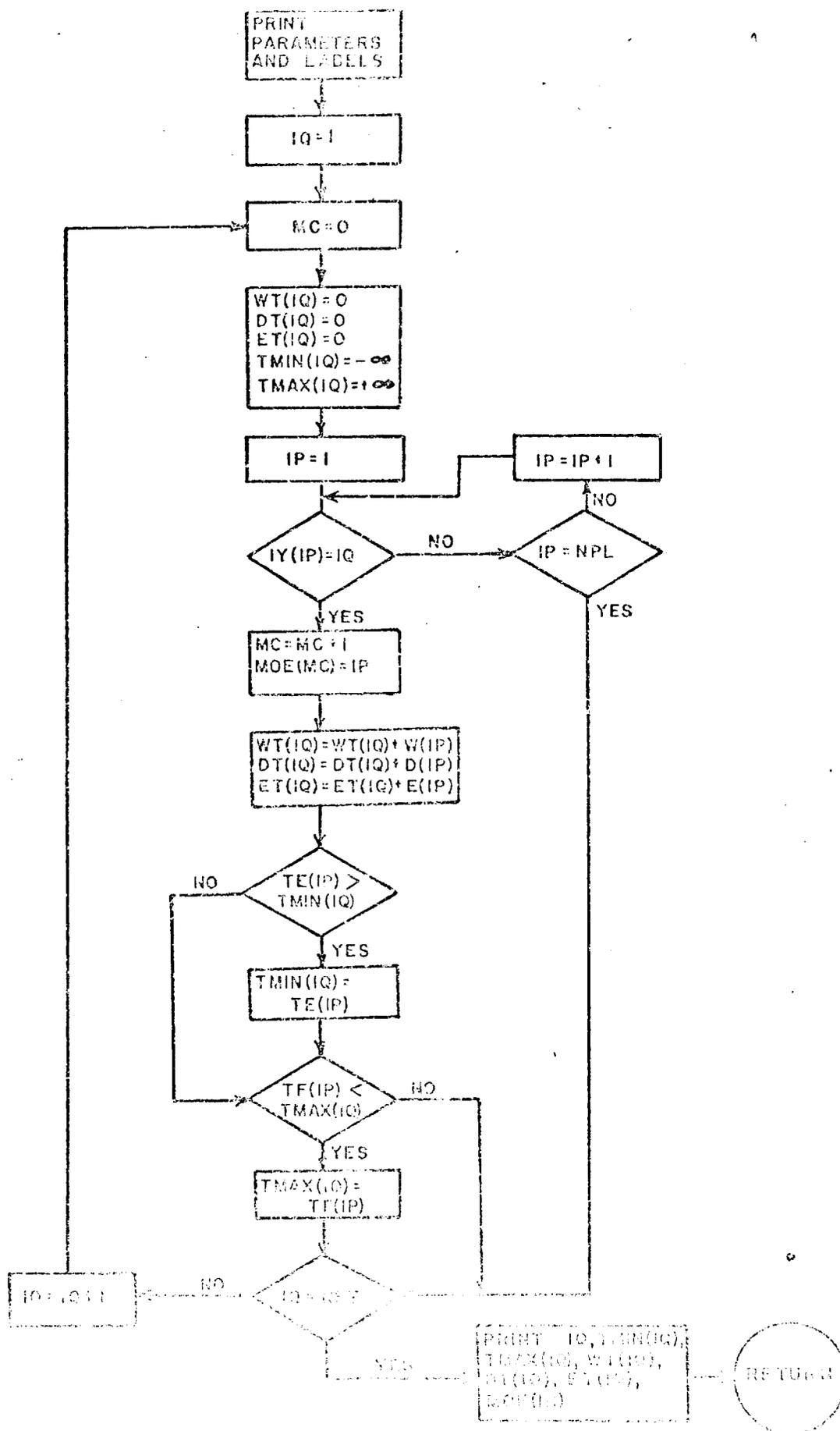
IP - Indexes the modules.

IR - Indexes the modules on a particular vehicle for printout.

The array not in common is:

MØE(IR) - this array is used to collect the individual modules on a vehicle for printing out.

## SUBROUTINE PRINT FLOW DIAGRAM



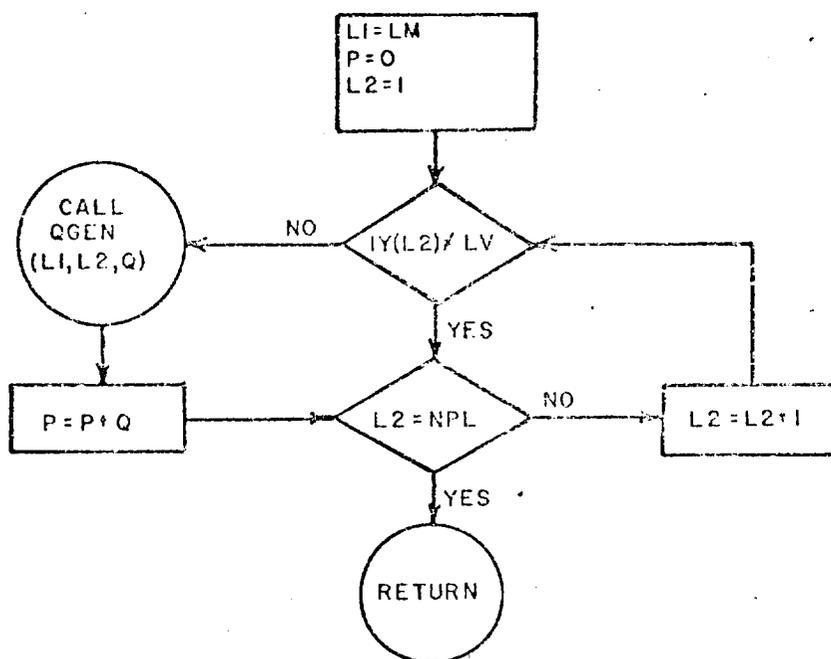
Subroutine PARGEN(LM, IV, P)

The purpose of this subroutine is to evaluate

$$\phi_{LM, LV}(\rho) = \sum_{L2 \in S_{LV}} q_{LM, L2}$$

where  $S_{LV} = \{L2 \mid \rho(L2) = IV\}$ ,

and  $q_{LM, L2}$  = interference of modules LM and L2.

SUBROUTINE PARGEN FLOW DIAGRAM

The variables not in common are:

L1 - Used to transfer LM to subroutine QGEN.

LM - Module number transferred down to PARGEN.

P - Returns the value of  $\phi_{LM, LV}$ .

L2 - Indexes the modules on vehicle IV.

IV - Vehicle number transferred down to PARGEN.

Q - Used to return the value of  $q_{L1, L2}$ .

Subroutine QGEN(L1,L2,Q)

The purpose of this subroutine is to supply  $q_{L1,L2}$  the interference of modules L1 and L2. The time intervals for the modules are:

$$T1 = [TE(L1), TF(L2)] \quad \text{and} \quad T2 = [TE(L2), TF(L2)] .$$

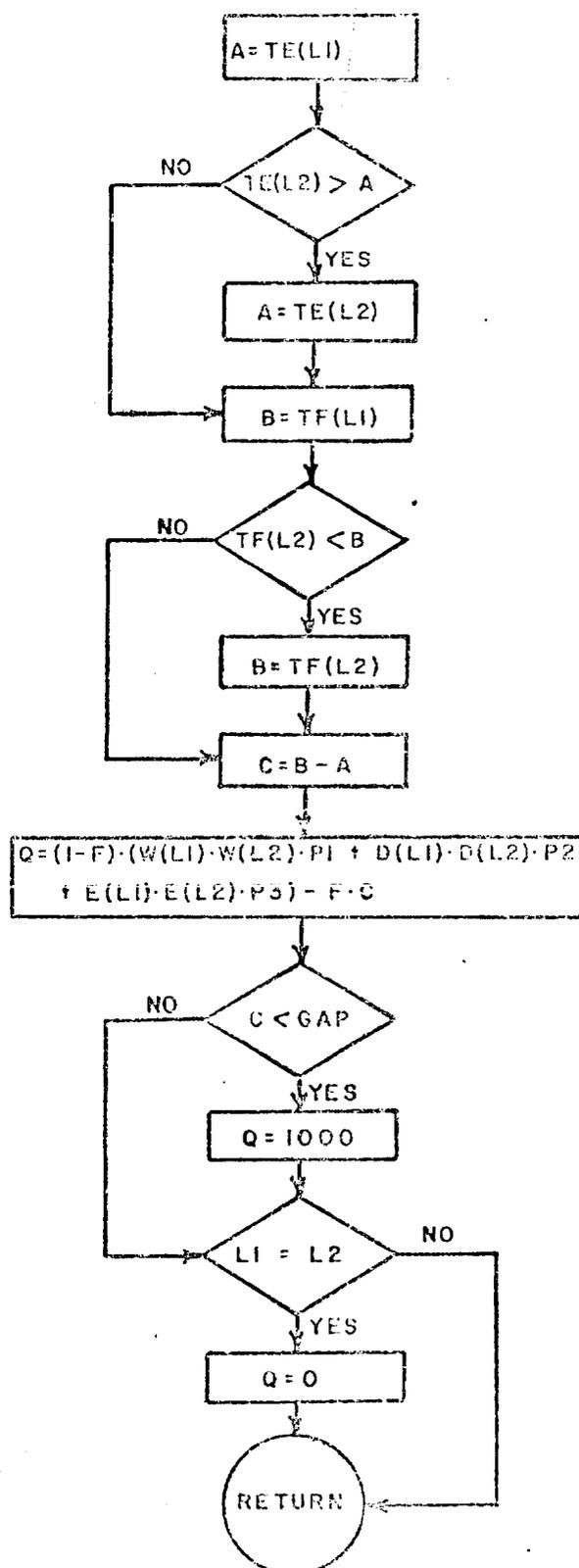
The interval of intersection is calculated as:

$$A = \max [TE(L1), TE(L2)] ; B = \min (TF(L1), TF(L2)) ; C = B - A .$$

The parameter GAP defines the specified overlap required and F defines the parametric weight assigned to the time interval as opposed to the physical loading characteristics. The parameters  $p_1, p_2$  and  $p_3$  assign relative weights to the physical loading characteristics of weight, diameter and length. So,

$$\left\{ \begin{array}{l} Q = (1-F) \cdot [p_1 \cdot w(L1) \cdot w(L2) + p_2 \cdot D(L1) \cdot D(L2) \\ \quad \cdot p_3 E(L1) \cdot E(L2)] - F \cdot C \quad \text{for} \left\{ \begin{array}{l} C > GAP \\ L1 \neq L2 \end{array} \right. \\ \\ Q = + \infty \quad (\text{taken as } 1000) \quad \text{for} \left\{ \begin{array}{l} C < GAP \\ L1 \neq L2 \end{array} \right. \\ \\ Q = 0 \quad \text{for } L1 = L2 \end{array} \right.$$

## SUBROUTINE QGEN FLOW DIAGRAM



The variables not in common are:

L1 - Module number transferred down to QGEN  
 L2 - Module number transferred down to QGEN

A -  $\max [TE(L1), TE(L2)]$

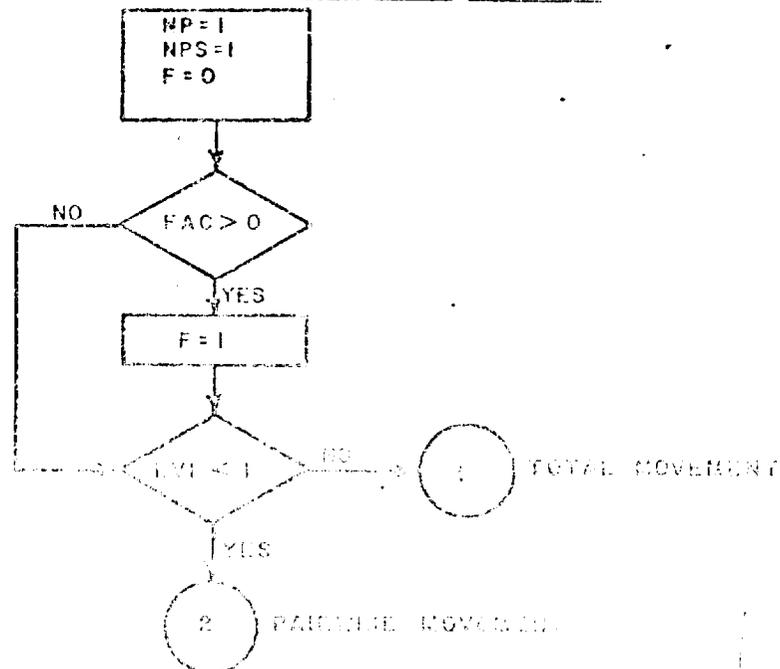
B -  $\min [TF(L1), TF(L2)]$

C - time interval  $(b-a)$

Q - used to return the value of  $u_{L1, L2}$

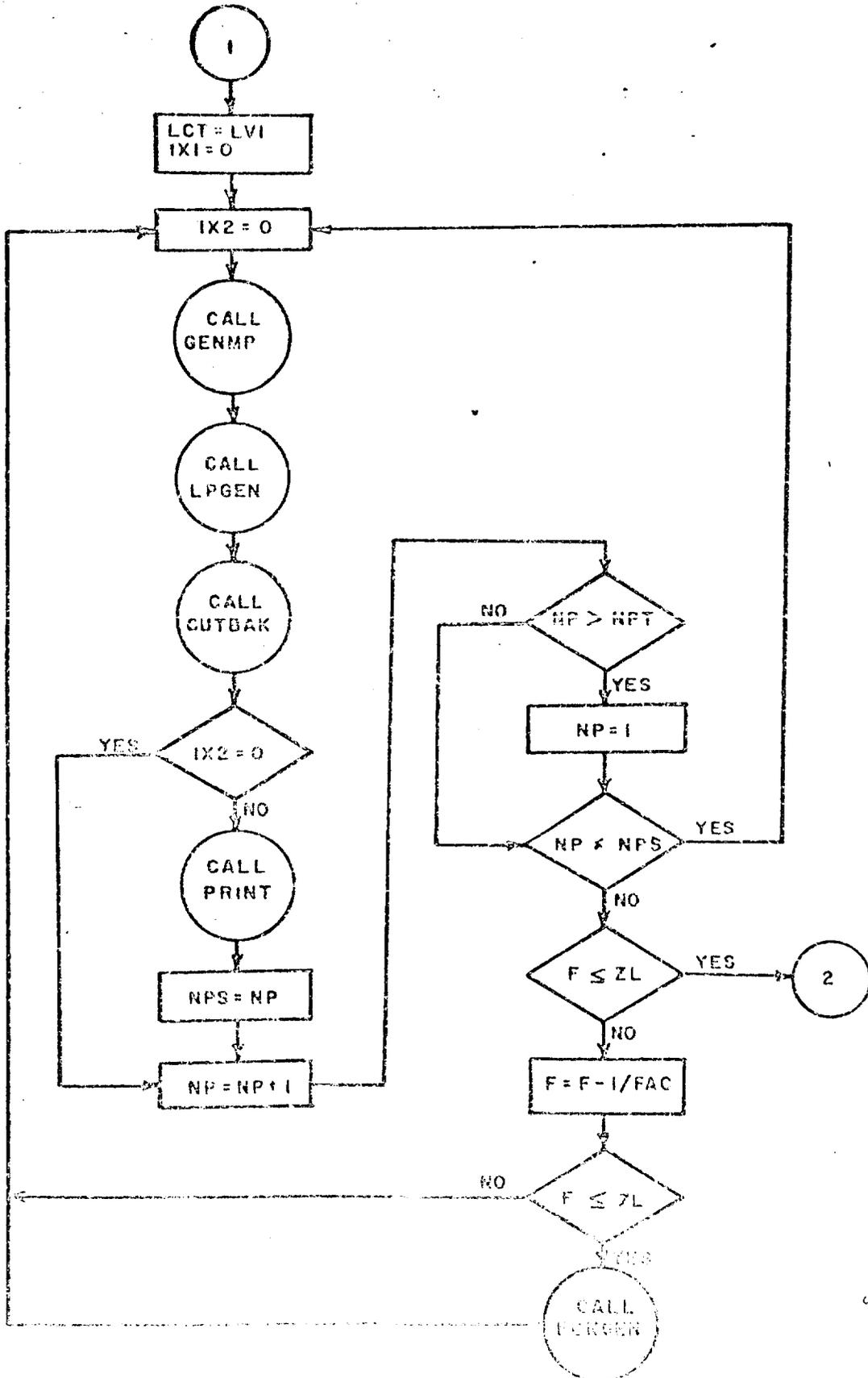
Subroutine SETUP

The principal purpose of this subroutine is to control the cycling through the partitioned subproblems. The extraction of the modules in a particular priority class is accomplished in subroutine GENMP which sets MAPP(IP) a list of the modules in the current priority class. The possible assignments are then recorded in a pair of vectors MAPR(i) and MAPC(i) and the potential switches are kept track of in the vectors J(i) and V(i). All these vectors are setup in subroutine LPGEN. The actual trial switches up through the levels LV1 for total vehicle movement and LV2 for pairwise vehicle movement is accomplished in subroutine CUTBAK. Each priority class is considered sequentially in turn until a complete cycle through all priority classes is accomplished without gain. In the pairwise vehicle movement all pairs of vehicles are cycled through until no gain is accomplished before moving on through the cycle of priority classes. The parametric adjustment from complete emphasis on time overlap to complete emphasis on physical loading characteristics in the number of steps specified by FAC is controlled in this subroutine.

SUBROUTINE SETUP FLOW DIAGRAM

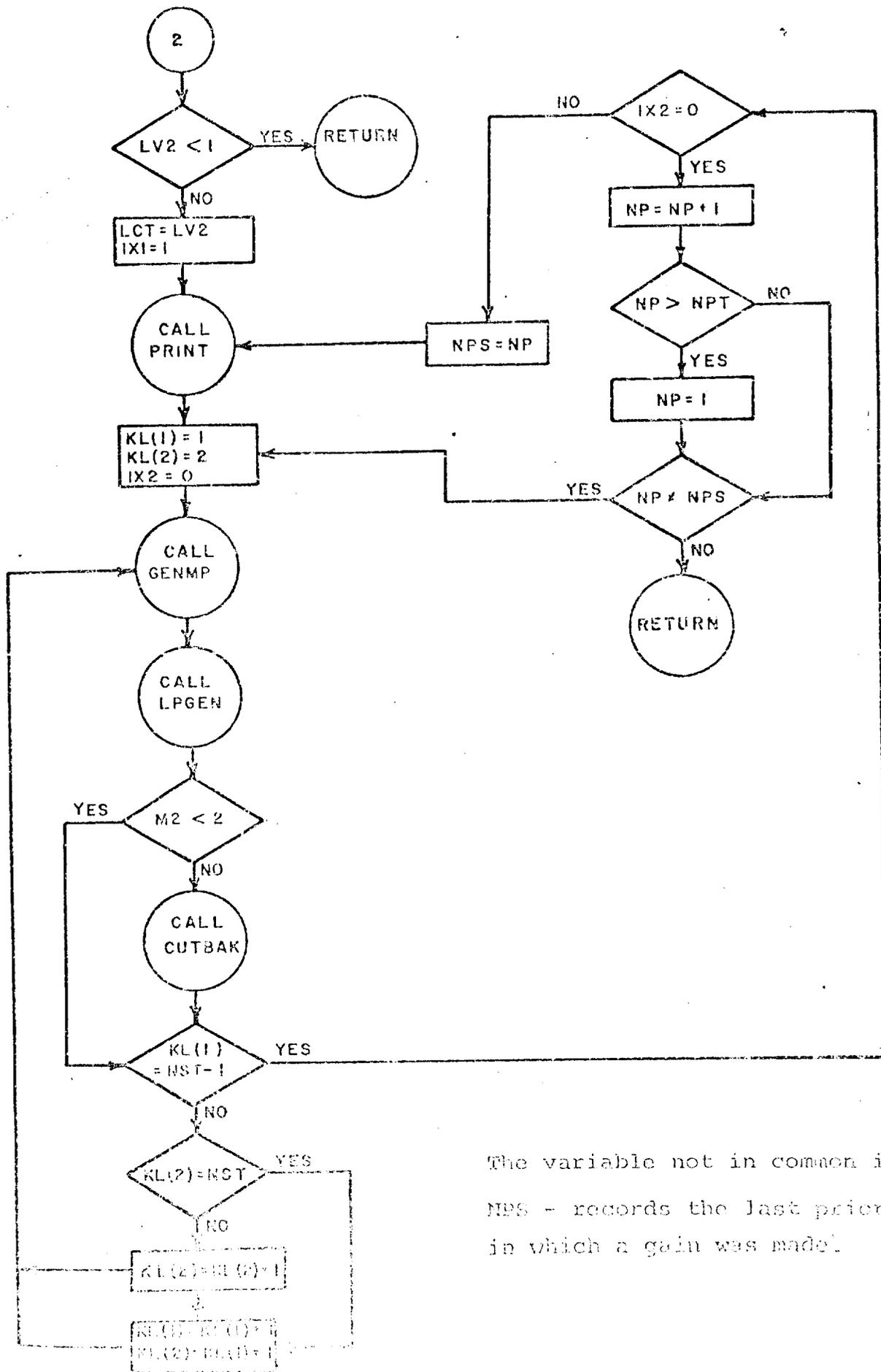
SUBROUTINE SETUP FLOW DIAGRAM (CONTINUED)

(Total Vehicle Movement)



SUBROUTINE SETUP FLOW DIAGRAM (CONTINUED)

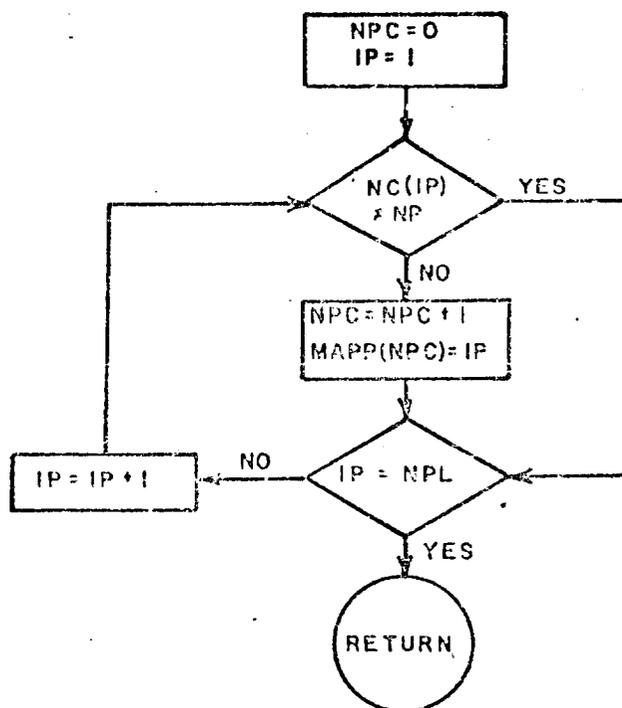
(Pairwise Vehicle Movement)



The variable not in common is:  
 NPS - records the last priority  
 in which a gain was made.

Subroutine GENMP

The purpose of this subroutine is to compile the list of modules in priority class NP and record them in the vector MAPP(i). The number of modules in the priority class NP is recorded in the variable NPC.

SUBROUTINE GENMP FLOW DIAGRAM

The variable not in common is:

IP - Indexes the modules.

subroutine LPGEN

The purpose of this subroutine is to setup the current subproblem for the investigation of potential interference reducing switches in assignments. The essential information of the investigation of these switches in assignment is contained in the four vectors MAPR(i), MAPC(i), J(i) and V(i). The two vectors MAPR(i) and MAPC(i) together record a linear list of all possible assignments within the current subproblem. The vector MAPR(i) records each module number in the current priority class that can be moved in the current subproblem repeated once for each vehicle to which it can be assigned. The vector MAPC(i) simply records the corresponding vehicle number. The vector J(k) is an index list of all possible assignments. It is used to keep track of the current trial assignments. Suppose there are M2 moveable modules in the current subproblem and NV possible assignments in the current subproblem. The index vector J(k) is always permuted so that the first M2 numbers index the current trial assignments. (The actual assignments are obtained from MAPR(J(k)) and MAPC(J(k)) for any J(k).) The potential switches are always indexed in the remaining (NV - M2) numbers in J(k). More precisely, each moveable module appears once and only once in the list MAPR(J(k)),  $k=1, \dots, m_2$ . Consider any module  $\text{MAPR}(J(k_1))=LM$ , with  $k_1 \leq m_2$ , which is currently assigned to the vehicle  $\text{MAPC}(J(k_1))=LV_1$ . Then any  $k_2 > m_2$  such that  $\text{MAPR}(J(k_2))=LM$  gives an alternate feasible assignment of LM to the vehicle  $\text{MAPC}(J(k_2))=LV_2$ . A switch in assignment is recorded by permutating the J(k) index list,

$$J'(k_2)=J(k_1) \quad \text{and} \quad J'(k_1)=J(k_2) \quad .$$

The change in the value of the interference functional for any such switch in assignment is carried in,

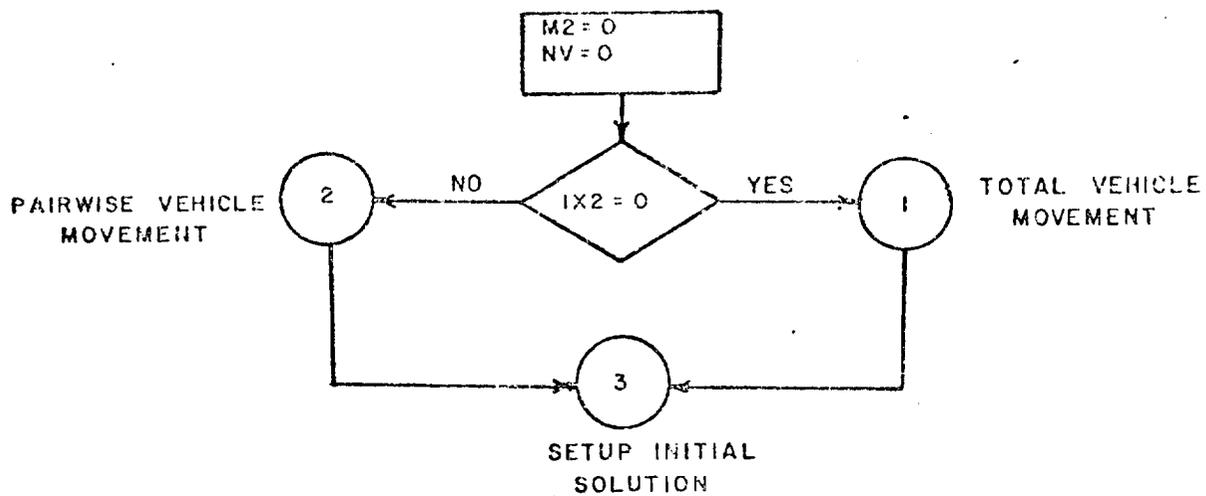
$$V(k_2) = \phi_{LM, LV_2} - \phi_{LM, LV_1}, \quad k_2 > m_2$$

For facility in updating the  $V(k)$  vector after such switches,

$$V(k_1) = -\phi_{LM, LV_1}, \quad k_1 \leq m_2.$$

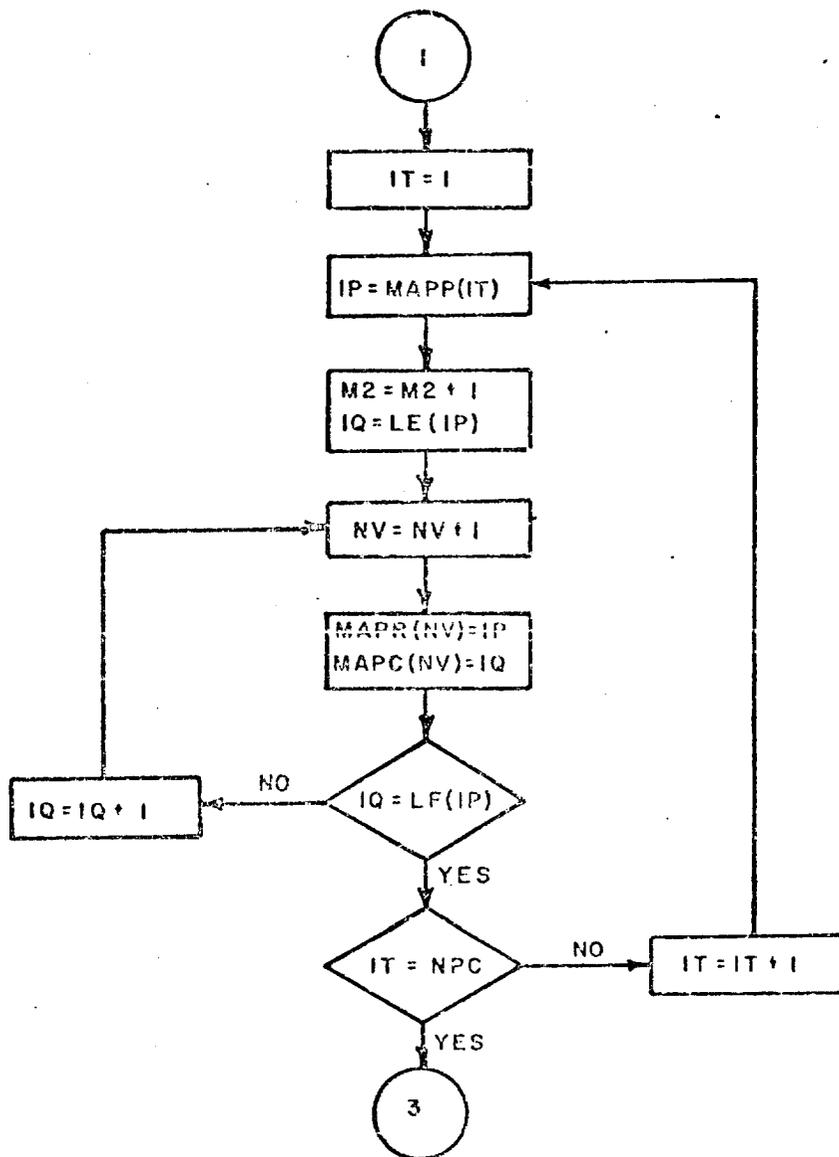
The transformation of the  $V(k)$  vector is accomplished in subroutine TRANS and will be discussed in connection with that subroutine. The actual search for gains in the interference functional is done in subroutine CUTBAK and will be discussed in that subroutine.

This subroutine LPGEN consists of three principal logical parts. The first part generates the subproblem when the modules are free to move to any feasible vehicle. The second part generates the problem when the modules are restricted to move in exchanges between pairs of vehicles. The third part sets up the initial solution in accordance with the current assignments.

SUBROUTINE LPGEN FLOW DIAGRAM

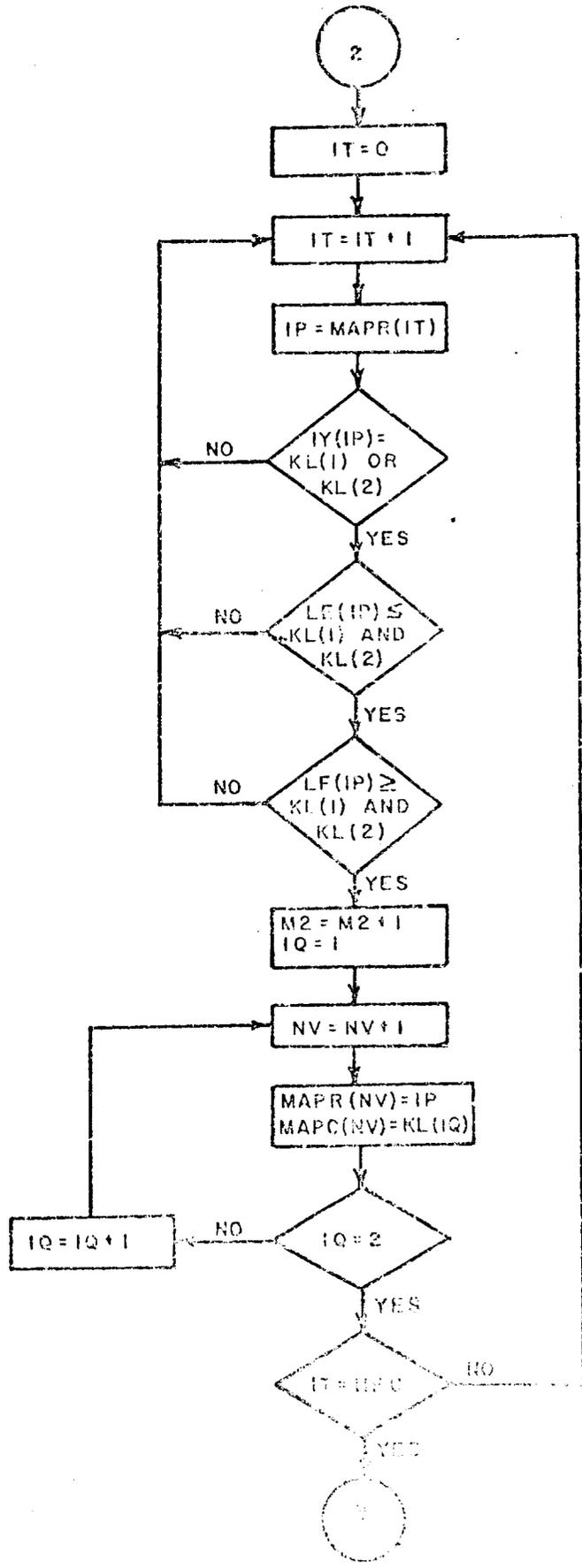
SUBROUTINE LPGEN FLOW DIAGRAM (CONTINUED)

(Total Vehicle Movement)



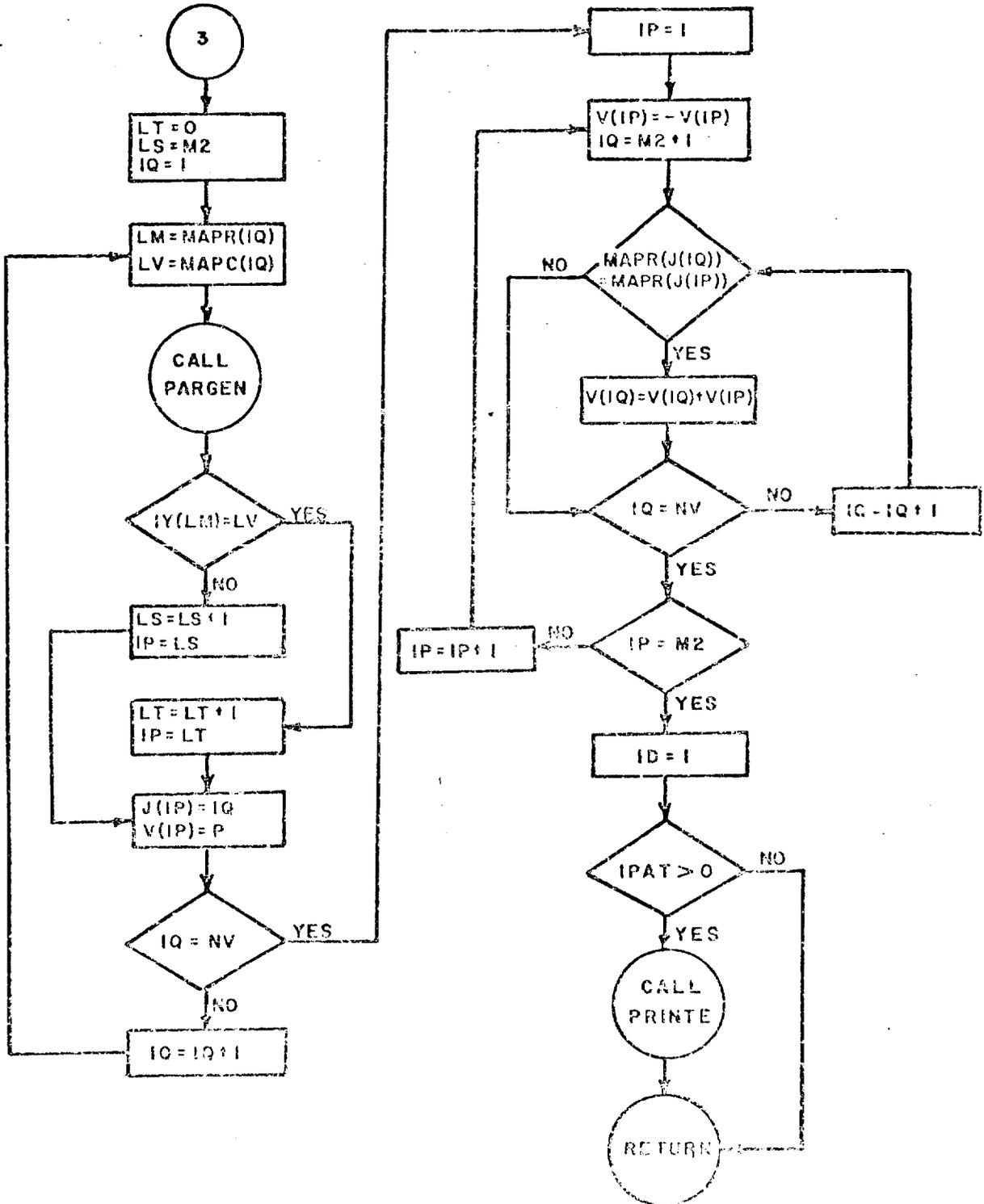
SUBROUTINE LPGEN FLOW DIAGRAM (CONTINUED)

(Pairwise Vehicle Movement)



SUBROUTINE LPGEN FLOW DIAGRAM (CONTINUED)

(Setup Initial Solution)



The variables not in common are:

- IT - Indexes modules in priority class
- IP - module numbers MAPP(IT)
- IQ - Indexes assignments
- IP1 - LE(IP)
- IP2 - LF(IP)
- IM - MAPR(IQ)
- IN - MAPC(IQ)
- P - Used to return the value of  $\phi_{LM, LV}$
- IM - M2+1
- JIQ - J(IQ)
- JIP - J(IP)

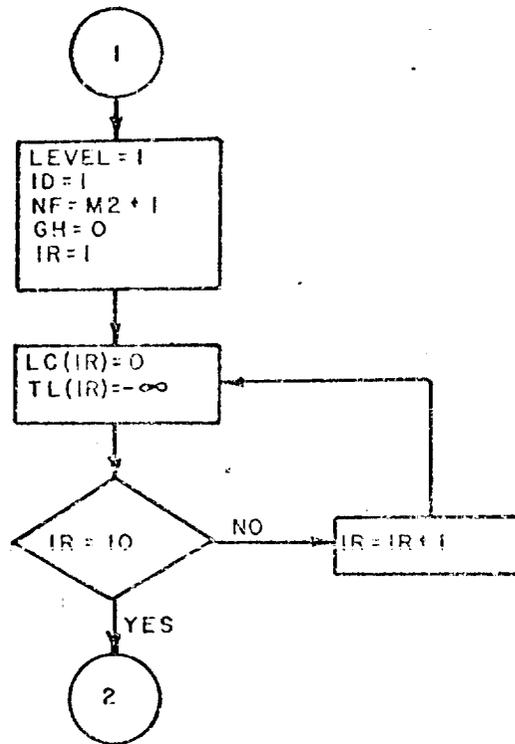
Subroutine CUTBAK

The purpose of this subroutine is to carry out the search for switches in assignments that will reduce the interference functional. The variable LEVEL records the current level of the search. The variable LEVEL starts with the value  $LEVEL=1$ . After all single switches of assignment have been implicitly enumerated, the value of LEVEL is incremented to  $LEVEL=2$ . When all double switches of assignment have been checked it is incremented again until the maximum  $LEVEL=LCT$  has been checked. The variable ID indexes the current number of switches made on the way up to LEVEL.

The variables  $TL(ID)$ ,  $LR(ID)$ ,  $LC(ID)$  record the history of the switches already made. All switches are tried in order of their change value  $V(IC)$ . The variable  $TL(ID)$  records the value of the switch  $V(ID)$  made at step  $ID$ . Suppose  $V(LS)$  for  $LS > M2$  is the next best value to be tried. The index  $LS$  of the trial assignment being switched in is recorded in  $LC(ID)=LS$ . There is then a unique current assignment occurring for some  $LT \leq M2$  such that  $MAPR(J(LS))=MAPR(J(LT))$ . This merely reflects the fact that each module must be currently assigned. The index  $LT$  of the assignment being switched out is recorded in  $LR(ID)=LT$ . The vector  $J(IC)$  is permuted  $J'(LS)=J(LT)$  and  $J'(LT)=J(LS)$  to record the switch in assignments in subroutine TRANS. The vector of change values  $V(IC)$  is also updated to reflect the change of assignment in subroutine TRANS and the actual formulas for the transition will be given in the discussion of subroutine TRANS. The history of the switches must be recorded in order to insure all unique combinations are tried without duplications. The detailed mechanics of the enumeration scheme are embodied in the following Flow Diagram.

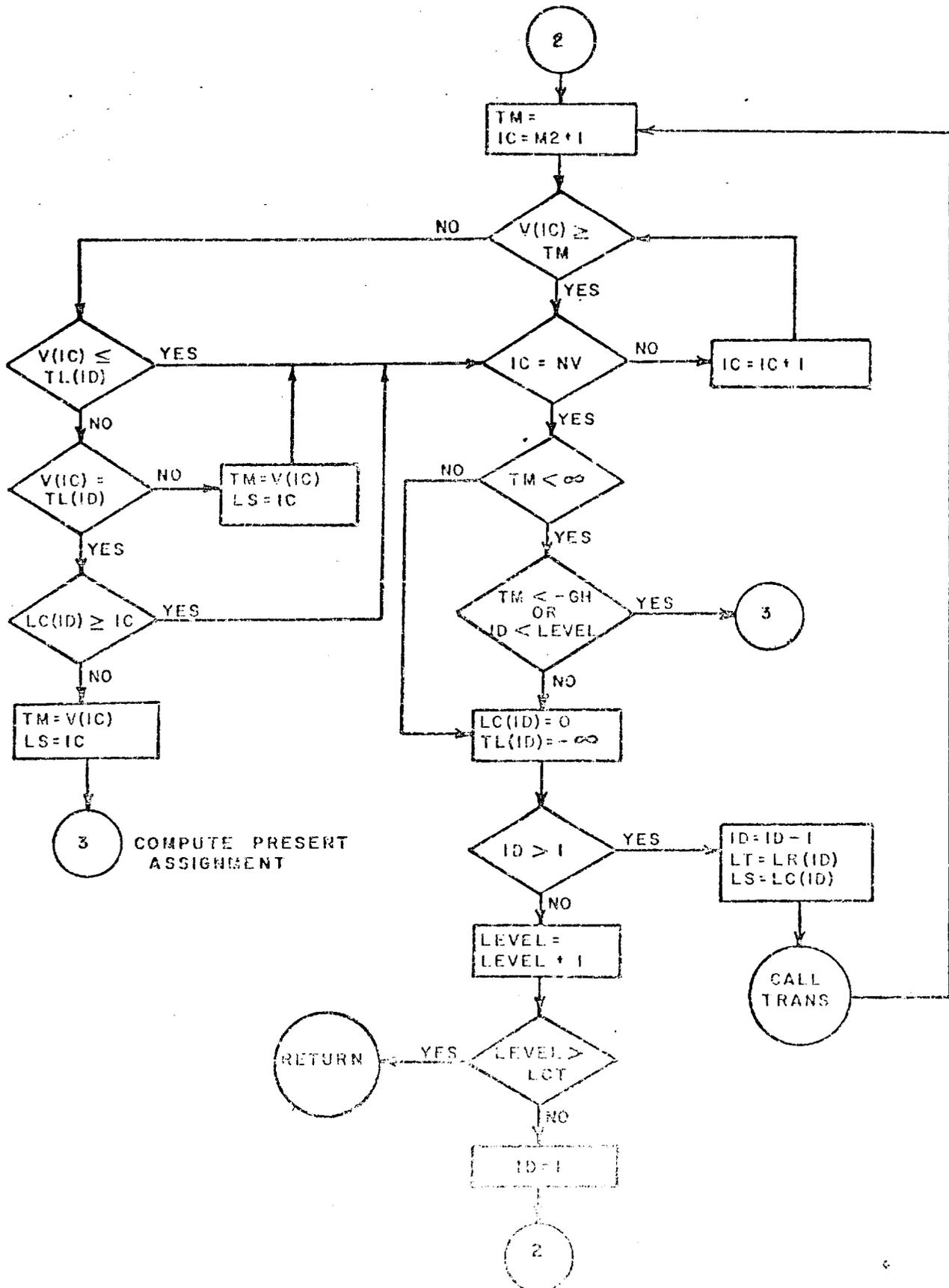
SUBROUTINE CUTBAK FLOW DIAGRAM

(Initialize Parameters)

COMPUTE INDEX OF NEW  
ASSIGNMENT LS

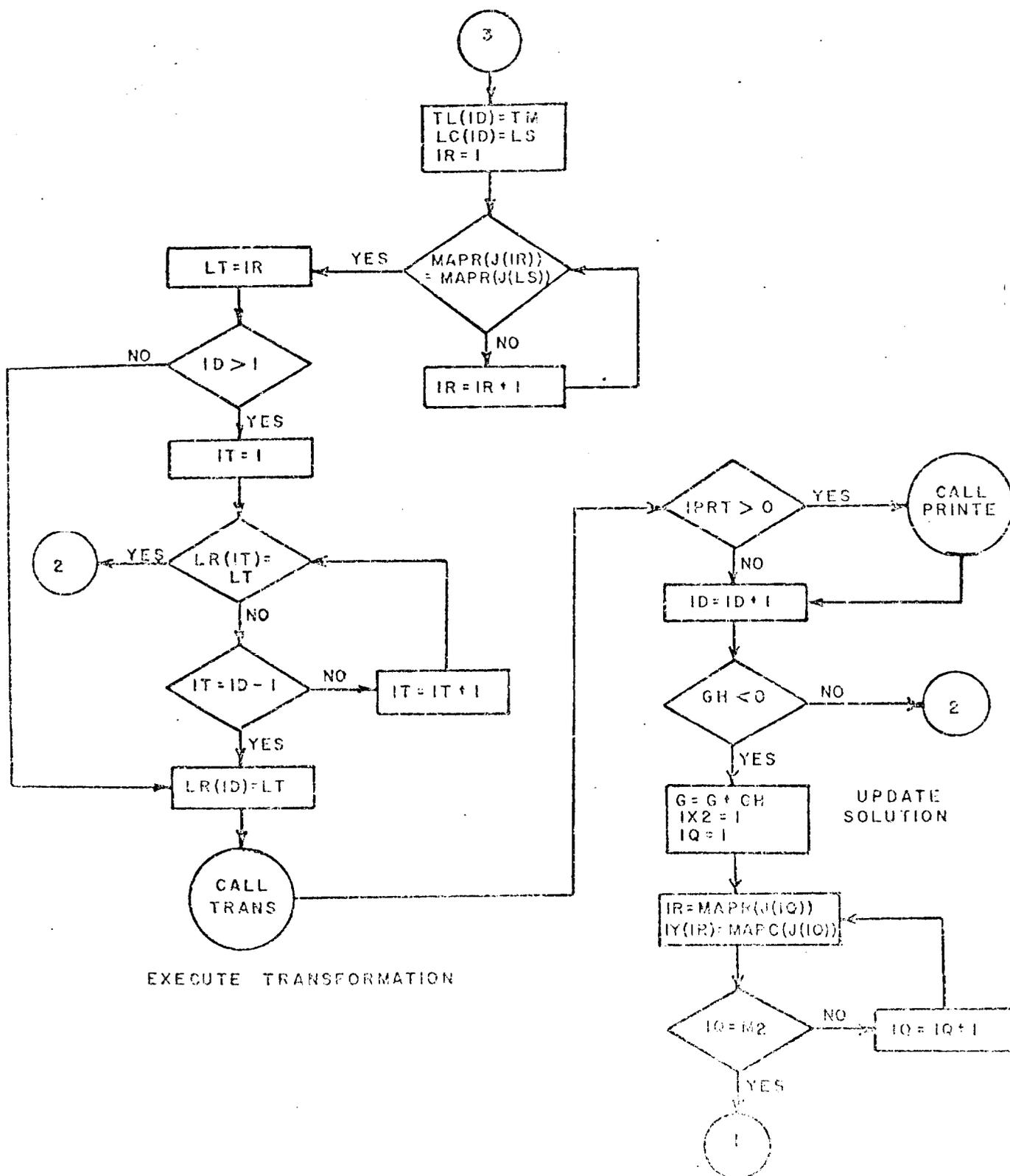
SUBROUTINE CUTBACK FLOW DIAGRAM (CONTINUED)

(Compute Index of New Assignment)



## SUBROUTINE CUTBAK FLOW DIAGRAM (CONTINUED)

(Compute Present Assignment Index  
Execute Transformation - Update Solution)



The variables not in common are:

- IR - Indexes assignments J(IR). (Also see below)
- NF - First potential assignment not being used  $NF=M2+1$
- TM - Best change value  $(\varphi_{jt}-\varphi_{js})$  not yet tried
- IC - Indexes gains V(IC)
- JLS - J(LS)
- JIR - J(IR)
- IN2 - ID-1
- IT - Indexes current assignments switched out LR(IT)
- IQ - Indexes assignments J(IQ)
- JIQ - J(IQ)
- IR\* - Also used for module numbers  $IR=MAPR(JIQ)$

The arrays not in common are:

- LR(ID) - Records index of a current assignment switched out to check for higher order gain.
- LC(ID) - Records index of a new trial assignment switched in to check for higher order gain.

Subroutine PRINTE

The purpose of this subroutine is to print "state of the solution" information. It is only used in diagnostic runs to locate problems within the computer code. The parameter value  $IPRT=1$  causes this subroutine to be called in  $LPGEN$  after the initialization of the problem, and in  $CUTBAK$  after each transformation of assignments. The first line of output gives  $G$  the current value of the interference functional and  $CH$  the incremental value of the current sequence of trial switches. The second line of output is the label "ASSIGNMENTS". This line is followed by the basic controlling vectors:

```
MAPR(IP) ; IP=1,...,NV
MAPC(IP) ; IP=1,...,NV
MAPF(IP) ; IP=1,...,NPC
```

The next line gives in labeled form the parameters:

```
M2 -- the current number of moveable modules
IX1 - indicates at the value  $IX1=1$  that currently the movement
      is restricted to pairwise vehicles
IX2 - indicates at value  $IX2=0$  that no gain has yet been made
      in the current priority class; indicates at value  $IX2=1$ 
      that a gain has been achieved in the current priority
      class
ID - indicates current level of trial switches being investi-
     gated
IP - index of assignment being switched out
IO - index of assignment being switched in
LEVEL- indicates current level of combinations of switches being
       investigated.
```

The next line is the label "J, V Vectors". This is immediately followed by the vectors:

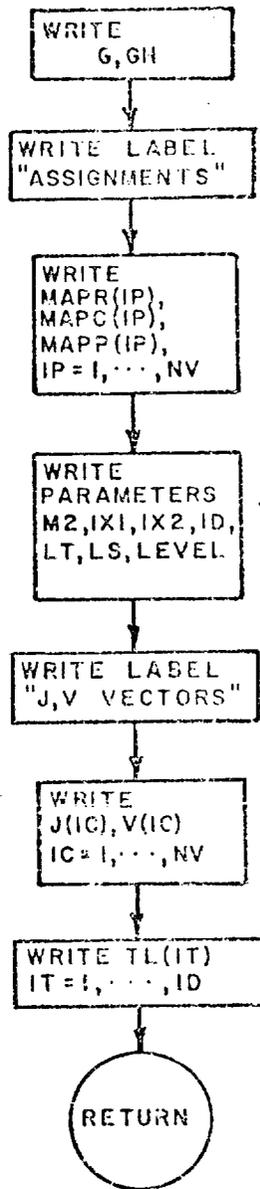
J(IC) ; ic=1,...,NV

V(IC) ; ic=1,...,NV .

This is followed by the labeled vector,

TL(IT) ; IT=1,...,ID

which gives the change values V(IC) used at the  $IT^{\text{th}}$  trial switch of assignment.

SUBROUTINE PRINTF FLOW DIAGRAM

The variables not in common are:

IP - Index for printing maps

IC - Index for printing J(IC) and V(IC)

IT - Index for printing TL(IT)

The format statements used in subroutine PRINTE are:

20 FØRMAT(1H,I6,9I12)

21 FØRMAT(1H,10F12.6)

34 FØRMAT(3HØG=E16.8,4H GH=E16.8)

36 FØRMAT(12H ASSIGNMENTS)

42 FØRMAT(1H 39I3)

4 FØRMAT(4H M2=I3,5H IX1=I3,5H IX2=I3,4H ID=I3,4H LT=I3,4H  
LS=I3,7H LEVEL=I3)

12 FØRMAT(' J,V VECTØRS')

19 FØRMAT('TL=',10F12.6)

The variables G and GH are printed with 34 FØRMAT.

The maps MAPR(IP), MAPC(IP) and MAPP(IP) are printed with 42 FØRMAT.

The parameters M2, IX1, IX2, ID, LT, LS, LEVEL are printed with  
4 FØRMAT.

The vector J(IC) is printed with 20 FØRMAT.

The vector V(IC) is printed with 21 FØRMAT.

The vector TL(IT) is printed with 19 FØRMAT.

Subroutine TRANS

The purpose of this subroutine is to carry out the transformation of the vectors  $J(IC)$  and  $V(IC)$  when a switch in assignment is made. The vectors are constructed so that for any  $IC$  in the range  $1 \leq IC \leq NV$ , the module is

$$L = \text{MAPR}(J(IC))$$

and the vehicle is

$$C = \text{MAPC}(J(IC))$$

The transition value vector  $V(IC)$  is constructed in such a manner that for,

any  $IC \leq M2$

$$L = \text{MAPR}(J(IC)) \text{ , } C1 = \text{MAPC}(J(IC))$$

then

$$V(IC) = -\phi_{L,C1}$$

any  $IC > M2$

such that

$$\text{MAPR}(J(IC)) = L \text{ , } \text{MAPC}(J(L)) = C2$$

then

$$V(IC) = \phi_{L,C2} - \phi_{L,C1}$$

Now consider any  $LT \leq M2$  with

$$L1 = \text{MAPR}(J(LT))$$

$$LTC = \text{MAPC}(J(LT))$$

$$V(LT) = -\phi_{L1,LTC}$$

and an  $LS > M2$

such that

$$\text{MAPR}(J(LS)) = L1$$

Let,

$$\text{MAPC}(J(LS)) = LSC$$

then

$$V(LS) = \varphi_{L1, LSC} - \varphi_{L1, LTC}$$

Now suppose the assignment  $L1 \rightarrow LTC$  is switched to  $L1 \rightarrow LSC$ .

Recall the transition formulas

$$\begin{aligned} \varphi'_{L,C} &= \varphi_{L,C} & C \neq LTC \text{ or } LSC \\ \left\{ \begin{array}{l} \varphi'_{L1, LTC} = \varphi_{L1, LTC} \\ \varphi'_{L1, LSC} = \varphi_{L1, LSC} \end{array} \right. & \text{since } \varphi_{L1, L2} = 0 \\ \left\{ \begin{array}{l} \varphi'_{L2, LTC} = \varphi_{L2, LTC} - \varphi_{L1, L2} \\ \varphi'_{L2, LSC} = \varphi_{L2, LSC} + \varphi_{L1, L2} \end{array} \right. & \text{for } L1 \neq L2 \end{aligned}$$

The effect on the transition value vector is now easily calculated as follows.

Basic Transformation

This transformation gives the effect for any IC such that  $\text{MAPR}(J(IC)) = L1$ .

$$V'(LT) = -\varphi'_{L1, LSC} = -\varphi_{L1, LSC} = V(LT) - V(LS)$$

$$V'(LS) = \varphi'_{L1, LTC} - \varphi'_{L1, LSC} = \varphi_{L1, LTC} - \varphi_{L1, LSC}$$

$$= -V(LS)$$

and for any other IC such that  $\text{MAPR}(J(IC)) = L1$  and  $\text{MAPC}(J(IC)) = C$  with  $C \neq LTC$  or  $LSC$ ,

$$V'(IC) = \varphi'_{L1, C} - \varphi'_{L1, LSC} = \varphi_{L1, C} - \varphi_{L1, LSC} = V(IC) - V(LS)$$

Secondary Transformation

This transformation gives the effect for any IC such that  $\text{MAPR}(J(\text{IC})) = L2$  and  $L2 \neq L1$ .

Consider the following exhaustive cases.

Case 1  $\text{IC} < M2$  (Current assignments)

a)  $\text{MAPC}(J(\text{IC})) = \text{LTC}$

$$\begin{aligned} V'(\text{IC}) &= -\phi'_{L2, \text{LTC}} = -(\phi_{L2, \text{LTC}} - q_{L1, L2}) \\ &= V(\text{IC}) + q_{L1, L2} \end{aligned}$$

b)  $\text{MAPC}(J(\text{IC})) = \text{LSC}$

$$\begin{aligned} V'(\text{IC}) &= -\phi'_{L2, \text{LSC}} = -(\phi_{L2, \text{LSC}} + q_{L1, L2}) \\ &= V(\text{IC}) - q_{L1, L2} \end{aligned}$$

Case 2  $\text{IC} > M2$  (Not current assignments)

Assume current assignment is  $L2 \rightarrow C$

a)  $\text{MAPC}(J(\text{IC})) = \text{LTC}$  and  $C \neq \text{LTC}$  or  $\text{LSC}$

$$\begin{aligned} V'(\text{IC}) &= \phi'_{L2, \text{LTC}} - \phi'_{L2, C} \\ &= (\phi_{L2, \text{LTC}} - q_{L1, L2}) - \phi_{L2, C} \\ &= V(\text{IC}) - q_{L1, L2} \end{aligned}$$

b)  $\text{MAPC}(J(\text{IC})) = \text{LSC}$  and  $C \neq \text{LTC}$  or  $\text{LSC}$

$$\begin{aligned} V'(\text{IC}) &= \phi'_{L2, \text{LSC}} - \phi'_{L2, C} \\ &= (\phi_{L2, \text{LSC}} + q_{L1, L2}) - \phi_{L2, C} \\ &= V(\text{IC}) + q_{L1, L2} \end{aligned}$$

c) MAPC(J(IC))=LSC and C=LTC

$$V'(IC) = \varphi'_{L2, LSC} - \varphi'_{L2, LTC}$$

$$= (\varphi_{L2, LSC} + q_{L1, L2}) - (\varphi_{L2, LTC} - q_{L1, L2})$$

$$= V(IC) + 2q_{L1, L2}$$

d) MAPC(J(IC))=LTC and C=LSC

$$V'(IC) = \varphi'_{L2, LTC} - \varphi'_{L2, LSC}$$

$$= (\varphi_{L2, LTC} - q_{L1, L2}) - (\varphi_{L2, LSC} + q_{L1, L2})$$

$$= V(IC) - 2q_{L1, L2}$$

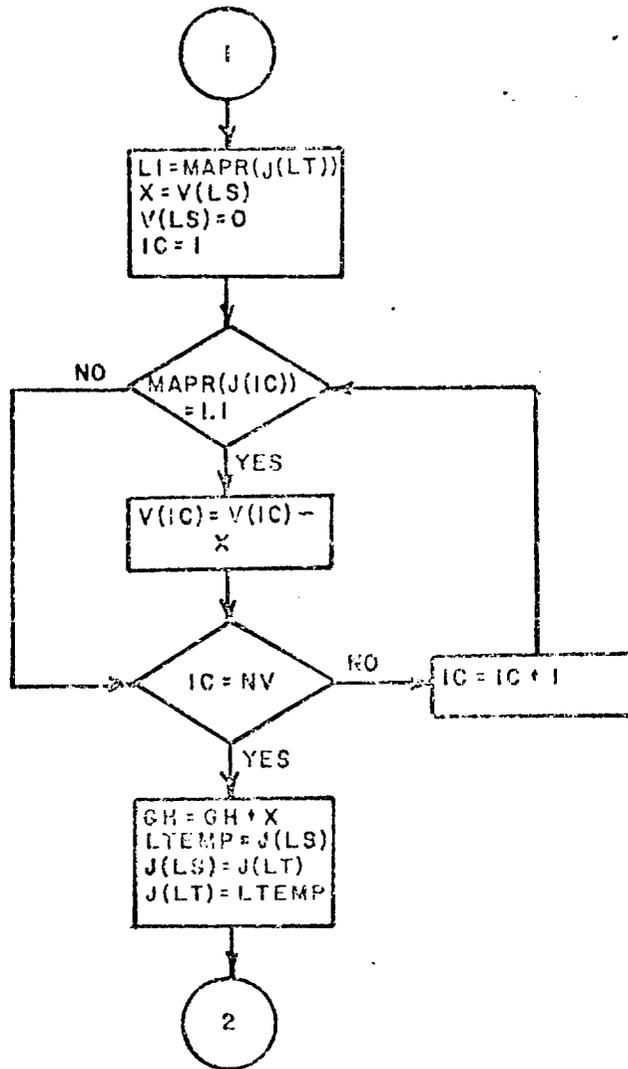
All other V'(IC) remain invariant!

The J(IC) vector is transformed as follows

$$J'(LT) = J(LS) \text{ and } J'(LS) = J(LT)$$

all other J'(IC) = J(IC).

SUBROUTINE TRANS FLOW DIAGRAM  
 (Basic Transformation)





The variables not in common are:

JLT - J(LT)  
L1 - MAPR(J(LT))  
X - V(LS)  
IC - Index of vectors  
JIC - J(IC)  
LTEMP- J(LS)  
JLS - J(LS)  
LTC - MAPC(J(LT))  
LSC - MAPC(J(LS))  
IP - Index of vectors  
JIP - J(IP)  
LZ - MAPR(J(IP))  
Q - - Returns value of  $q_{L1,L2}$   
IRT - MAPR(J(IC))  
ICT - MAPC(J(IC))

GLOSSARY

(Variables and Vectors in Common Statement)

Variables

- F - The parametric weight assigned to the time interval overlap
- FAC - The number of parametric steps to shift from complete emphasis on time interval feasibility to complete emphasis on weight-diameter-length balancing
- G - The value of the interference functional
- GAP - The scheduling time interval required for the flights. The final permissible time interval for each shuttle craft flight must be greater than or equal to GAP
- GH - The change value of the current sequence of trial switches
- ID - The variable ID indexes the current number of switches made on the way up to Level
- IPRT - Intermediate printout indicator parameter
- IX1 - Indicates at the value IX1=0 that the movement of the modules is over all the vehicles; at the value IX1=1, indicates the movement of the modules is restricted to pairwise vehicles
- IX2 - Indicates at the value IX2=0 that no gain has yet been achieved in the current subproblem; at the value IX2=1, that a gain has been achieved in the current subproblem
- LCT - The level count for the current subproblem. This variable assumes the values LV1 and LV2 depending on the current operating mode
- LEVEL - This variable is the current level of the search. It is the number of modules involved in the current switches being investigated
- LS - Index of the assignment being switched in given by  $J(LS)$

- LT - Index of the assignment being switched out given by  $J(LT)$
- LV1 - Level of interchanges searched when the modules are free to move to any other shuttle craft flight
- LV2 - Level of interchanges searched when the modules are restricted to interchange between pairs of shuttle craft flights
- M2 - The number of moveable modules in the current subproblem
- NP - The current priority class under scrutiny
- NPC - The number of modules in the current priority class
- NPL - The total number of modules in the current run
- NPT - The total number of priority classes
- NST - The total number of shuttle craft flights in the current run
- NV - The total number of possible assignments in the current subproblem
- P1 - The relative emphasis to be placed on weight balancing as opposed to diameter and length balancing. In general,  

$$P1+P2+P3=1 \text{ and } P1 \geq 0, P2 \geq 0, P3 \geq 0$$
- P2 - The relative emphasis to be placed on diameter balancing as opposed to weight and length balancing
- P3 - The relative emphasis to be placed on length balancing as opposed to weight and diameter balancing.

Vectors

- $V(IC)$  - This vector carries the change values of switches in assignment. Let  
 $L=MAPR(J(IC)), C=MAPC(J(IC))$   
 then for  
any  $IC \leq M2$   
 $V(IC) = -\phi_{L,C}$   
for any  $IC > M2$   
 such that  
 $MAPR(J(IC)) = L, MAPC(J(IC)) = C1$   
 $V(IC) = \phi_{L,C1} - \phi_{L,C}$
- $J(IC)$  - This vector is an index list of all possible assignments within a partitioned subproblem. It is always permuted so that the first  $m2$  numbers index the current trial assignments.
- $MAPR(IC)$  - This vector in conjunction with the vector  $MAPC(IC)$  records a linear list of all possible assignments within a subproblem. The vector  $MAPR(IC)$  records each moveable module number once for each vehicle to which it can be assigned in the current subproblem.
- $MAPC(IC)$  - This vector in conjunction with the vector  $MAPR(IC)$  records a linear list of all possible assignments within a subproblem. The vector  $MAPC(IC)$  records a vehicle number to which the module given in  $MAPR(IC)$  can be assigned.
- $W(IP)$  - The vector of weights of the modules
- $D(IP)$  - The vector of diameters of the modules
- $E(IP)$  - The vector of lengths of the modules
- $WT(IR)$  - The vector of total weight on a vehicle
- $DT(IR)$  - The vector of total diameter on a vehicle
- $ET(IR)$  - The vector of total length on a vehicle
- $TE(IP)$  - The earliest possible schedule time for a module

- TF(IP) - The latest possible schedule time for a module
- TMIN(IR) - The vector of first possible schedule times for the vehicles
- TMAX(IR) - The vector of last possible schedule times for the vehicles
- LE(IP) - The first vehicle to which a module can be assigned
- LF(IP) - The last vehicle to which a module can be assigned
- TL(ID) - The vector TL(ID) records the value of the switch V(ID) made at trial switch ID in the current sequence of trial switches
- IY(IP) - The vector of current assignments. IY(IP) gives the vehicle to which the module IP is assigned
- NC(IP) - The vector of priority class assignments of the modules

## APPENDIX I

## Review of Mathematical Terminology and Concepts.

In addition to the mathematics specifically required for linear programming, the following summary of the underlying concepts and terminology of modern mathematics should provide a useful orientation to the reader.

- A.1.1 The notions of element, set and relation can be taken as logical primitives about which everyone has some intuitive feel. A set is composed of elements capable of possessing certain properties and of having, between themselves or with elements of other sets, certain relations. A set is simply a well-defined collection of objects called "elements". We shall denote sets by capital letters. The elements will be enclosed by braces  $\{ \}$ . Sets are commonly described either explicitly or implicitly. With finite sets the elements may be enumerated, i.e.,  $A = \{2, 4, 6\}$ , the numbers two, four, and six. Infinite sets are described by giving a defining property of a generic element of the set. The defining property may be explicit or implicit; explicit properties indicate how any desired included element of the set may be generated, and implicit properties are those in which the defining property merely serves to test proffered candidates for inclusion or exclusion. The normal way of expressing the defining property takes the form  $\{x | P(x)\}$ , the set of elements  $x$  which have the property  $P(x)$ . For example,  $A = \{x | x \geq 0\}$ , the set of non-negative numbers;  $B = \{(x, y) | x + y \leq 5\}$ , the set of ordered pairs  $(x, y)$  of number whose sum does not exceed five. Sometimes the defining property is taken as obvious when a few of the elements are listed, such as  $\{1, 2, \dots, n\}$ , all natural numbers one through  $n$  inclusive;  $\{1/2, 1/4, 1/8, \dots\}$ , all fractions

of the form  $2^{-n}$ ,  $n=1, 2, \dots$ . Sometimes sets such as  $\{a_1, a_2, \dots, a_n\}$  are written as  $\{a_i\}$  if there is no ambiguity in the discussion as to which elements  $a_i$  we are referring.

- A.1.2 A letter may designate either a "fixed" element or an arbitrary element (also called "variable", argument or generic element) of a set. An arbitrary element in a relation is "given" or "takes" the values of a fixed element when it is replaced in the relation by the fixed element. A general notation for a relation is  $R\{x, y, z\}$  where  $x, y, z$  would be the elements which are involved in the relation. A relation involving variables is an identity if it becomes a true proposition for whatever values are given the variables. If  $R$  and  $S$  designate two relations (or properties) then we say  $R$  implies  $S$  (often written  $R \rightarrow S$ ) if  $S$  becomes true whenever we replace the variables in  $R$  in such a way as to cause it to be true. We also say  $R$  and  $S$  are equivalent when each relation implies the other.
- A.1.3 Given a relation  $R\{x, y, z\}$  between the variables  $x, y, z$ , the phrase  $\langle \text{for all } x, R\{x, y, z\} \rangle$  (or  $\langle \forall x, R\{x, y, z\} \rangle$ ) is a relation between  $y$  and  $z$  which is true for a system of values given to these latter variables and all values given to  $x$ . The phrase  $\langle \text{there exists } x \text{ such that } R\{x, y, z\} \rangle$  (or  $\langle \exists x, \exists R\{x, y, z\} \rangle$ ) is a relation between  $y$  and  $z$  which is true for a system of values given to  $y$  and  $z$  and at least one value of  $x$ . These ideas, of course, extend to an arbitrary number of variables. Examples of this usage might be statements such as:  $\langle \text{for all circles, area} = \pi r^2 \rangle$ , a relationship between circles, area and radius; or  $\langle \exists \text{ a positive integer } N, \exists (n > N \Rightarrow |1 - x_n| < c) \rangle$ . Taking  $\bar{R}$  as the negation of  $R$ , the negation of  $\langle \forall x, R\{x, y, z\} \rangle$  is  $\langle \exists x \exists \bar{R} \rangle$  and the negation of  $\langle \exists x, \exists R \rangle$  is  $\langle \forall x, \bar{R} \rangle$ .

If  $R, S$  designate two relations, we consider  $(R \text{ and } S)$  as a single relation which is true only when both parts  $R, S$  are true. Likewise  $(R \text{ or } S)$  is a single relation which is considered true when either of  $R, S$  are true (and in particular when both are true). The logical or therefore does not have the disjunctive sense sometimes encountered in ordinary usage. With  $\bar{R}, \bar{S}$  as the negative of  $R, S$ , the negation of  $(R \text{ and } S)$  is  $(\bar{R} \text{ or } \bar{S})$  and the negation of  $(R \text{ or } S)$  is  $(\bar{R} \text{ and } \bar{S})$ .

In logic when we write  $(a = b)$  (read equals), it means logical identity. The symbols  $a, b$  represent the same element. The negation of this relation ( $\neq$ ) means the elements are different.

A.1.4 Given a set  $E$ , and a property of a generic element of  $E$ , those elements of  $E$  which possess this property constitute a new set  $A$ , called a subset of  $E$ . The property of belonging to a set  $A$  is written  $x \in A$  and is read as  $x$  is an element of  $A$ . The negation of this property is written  $x \notin A$  and is read  $x$  is not an element of  $A$ . The subset of all such elements is called the complement of  $A$  and is written as  $(A \text{ or } \bar{A})$ . It is possible that a property (for instance  $x = x$ ) is true for all elements of  $E$ . Thus the set  $E$  is a subset of itself and is called the universal (or whole, or entire) subset. Likewise, certain properties (for instance  $x \neq x$ ) do not hold for any elements of  $E$ . Such a subset is called the null (or empty) set and is frequently written as  $\emptyset$ . (Note that  $E$  and  $\emptyset$  are complements). It should be stressed that different but equivalent properties give rise to the same subset.

It also occurs with any fixed element  $a \in E$  that certain properties (for instance  $x = a$ ) hold solely for this element. In this way one obtains the subsets  $\{a\}$  reduced to single elements.

A.1.5 The set of all subsets of a given set  $E$  is itself a set called the power set, and written  $P(E)$ . Observe that  $\emptyset \in P(E)$ ,  $E \in P(E)$  and  $\forall x \in E, \{x\} \in P(E)$ . If  $x$  designates a generic element of  $E$ , and  $X$  a generic element of the set  $P(E)$ , the relation  $\langle x \in X \rangle$  between  $x$  and  $X$  is called the relation of belonging.

A.1.6 Given two elements  $x$  and  $y$  of  $E$ , and a generic element  $X$  of  $P(E)$ , the relation of equality  $\langle x = y \rangle$  is equivalent to the relation  $\langle \forall x \exists x \in X, \text{ one has } y \in X \rangle$ . In other words, two elements are the same element logically if there does not exist at least one subset which contains one of the elements and not the other.

A.1.7 Let  $X, Y$  be two subsets of a set  $E$ . If the property  $x \in X$  implies  $x \in Y$ , or in other words, if all the elements of  $X$  also belong to  $Y$ , we say that  $X$  is contained in  $Y$ , or that  $Y$  contains  $X$ , or that  $X$  is a subset of  $Y$ . This relation between  $X$  and  $Y$  is called the relation of inclusion (of  $X$  in  $Y$ ), and is denoted as  $\langle X \subset Y \rangle$  or  $\langle Y \supset X \rangle$ . Its negation is denoted  $\langle X \not\subset Y \rangle$  or  $\langle Y \not\supset X \rangle$ . The following elementary inclusion properties hold for arbitrary generic subsets  $X, Y$ , and  $Z$  of  $E$ :

- 1)  $\emptyset \subset X$  and  $X \subset E$
- 2)  $X \subset Y$  and  $Y \subset Z$  implies  $X \subset Z$
- 3)  $X \subset Y$  and  $Y \subset X$  is logically equivalent to  $X = Y$
- 4) the relation of "belonging"  $x \in X$  is equivalent to the relation of "inclusion"  $\{x\} \subset X$ .

A.1.8 For any two subsets of  $E$ , the set of elements possessing the property that  $\langle x \in X$  or  $x \in Y \rangle$  is written  $X \cup Y$  and is called the union of  $X$  and  $Y$ ; the set of elements possessing the property  $\langle x \in X$  and  $x \in Y \rangle$  is written  $X \cap Y$  and is called the intersection of  $X$  and  $Y$ . These definitions can

be immediately extended to any number of sets. The sets  $\{x\} \cup \{y\} \cup \dots \cup \{z\}$  are also written as  $\{x, y, \dots, z\}$ . When  $X \cap Y \neq \emptyset$ , we say the sets meet, and when  $X \cap Y = \emptyset$ , we say the sets are disjoint.

The following propositions summarize some of the more important properties and relations in set theory. They hold for any subsets  $X, Y, Z$  of a set  $E$ .

- 1)  $\emptyset = \complement E$  and  $E = \complement \emptyset$ .
- 2)  $\complement(\complement X) = X$ .
- 3)  $X \cup X = X, X \cap X = X$ .
- 4)  $X \cup (\complement X) = E, X \cap (\complement X) = \emptyset$ .
- 5)  $X \cup \emptyset = X, X \cap E = X$ .
- 6)  $X \cup E = E, X \cap \emptyset = \emptyset$ .
- 7)  $X \cup Y = Y \cup X, X \cap Y = Y \cap X$ . (commutative)
- 8)  $X \subset X \cup Y, X \cap Y \subset X$ .
- 9)  $\complement(X \cup Y) = (\complement X) \cap (\complement Y), \complement(X \cap Y) = (\complement X) \cup (\complement Y)$ .
- 10)  $X \cup (Y \cup Z) = (X \cup Y) \cup Z = X \cup Y \cup Z,$   
 $X \cap (Y \cap Z) = (X \cap Y) \cap Z = X \cap Y \cap Z$  (associative).
- 11)  $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z),$   
 $X \cap (Y \cup Z) = (X \cap Y) \cup (X \cap Z)$  (distributive).

The relations  $X \subset Y, \complement X \supset \complement Y, X \cup Y = Y, X \cap Y = X$  are logically equivalent.

The relations  $X \cap Y = \emptyset, X \subset \complement Y, Y \subset \complement X$  are logically equivalent.

The relations  $X \cup Y = E, \complement X \subset Y, \complement Y \subset X$  are logically equivalent.

The relation  $X \subset Y$  implies the relations  $X \cup Z \subset Y \cup Z$  and  $X \cap Z \subset Y \cap Z$ .

The relation  $\langle Z \subset X \text{ and } Z \subset Y \rangle$  is equivalent to  $Z \subset X \cap Y$ ; the relation  $\langle X \subset Z \text{ and } Y \subset Z \rangle$  is equivalent to  $X \cup Y \subset Z$ .

A.1.9 The identities given in (9) furnish a powerful means of deducing true propositions in the algebra of sets. When a set  $A$  of  $E$  is defined in terms of other sets  $X, Y, Z$  of  $E$  solely by the application of the operations  $\zeta, U, \cap$  (in arbitrary order) then the complement  $\zeta A$  can be obtained by applying the identities in (9). This complement  $\zeta A$  is obtained by replacing the subsets  $X, Y, Z$  by their complements and interchanging the operations  $U, \cap$  with  $\cap, U$  respectively while respecting the order of the operations. This is the rule of duality. In practice given an equality  $A = B$  between subsets of the above form, the first step is to consider the equivalent equality  $\zeta A = \zeta B$ . These expressions are then written out explicitly using the rule of duality. Finally, since we are dealing with generic sets  $X, Y, Z$ , the role of set and complement set are interchanged. The result of this procedure is the dual of  $A = B$ . Expressions involving inclusion,  $A \subset B$ , can also be treated in the same manner but it is then necessary to change the sign  $\langle \subset \rangle$  to  $\langle \supset \rangle$ . The identities given in A.1.8 under the same number are dual to each other. For instance in A.1.8 (3) we have the relation  $X \cup X = X$ . Applying the above procedure we have  $A = X \cup X$  and  $B = X$ . Since in addition to  $A = B$  we also have  $\zeta A = \zeta B$ , we apply the identities in (9) and arrive at  $(\zeta X) \cap (\zeta X) = \zeta X$ . Now interchanging the role of set and complement set we re-write this relation  $X \cap X = X$  and reach the dual expression given in (3).

A.1.10 In some equations, it is desirable to focus attention on a fixed subset  $A$  of  $E$ . Given another arbitrary subset  $X$  of  $E$ , the subset  $X \cap A$  is called the trace of  $X$  on  $A$  and is often written as  $X_A$  and is always considered as a subset of  $A$ . For all subsets  $X, Y$  of  $E$ , the following expressions hold:

$$(X \cup Y)_A = X_A \cup Y_A$$

$$(X \cap Y)_A = X_A \cap Y_A$$

$$\zeta X_A = (\zeta X)_A$$

where  $\zeta_E X$  is the complement of  $X$  with respect to  $E$  and  $\zeta_A X_A$  is the complement of the set  $X_A$  with respect to  $A$ . In a similar manner, with  $\xi$  a family of subsets of  $E$ , the trace of  $\xi$  on  $A$  is written  $\xi_A$  and consists of the traces on  $A$  of all the subsets of  $\xi$ .

## APPENDIX II

## Functions

- A.2.1 Given two sets E and F which may be distinct, a relation between a variable x of E and a variable y of F is called functional in y when for all  $x \in E$  there exists one and only one element y of F which satisfies the given relation with x. The rule or process which associates with each  $x \in E$  an element  $y \in F$  is called a function. The element y is the value of the function for the element x and the function is determined by the functional relation under consideration. Two equivalent functional relations determine the same function. As an example,

$$y = 2\left[\frac{(x-1)}{(x+1)} + \frac{(x-1)^3}{3(x+1)^3} + \frac{(x-1)^5}{5(x+1)^5} + \dots\right]$$

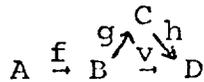
and

$$y = \int_1^x \frac{dt}{t}$$

are two equivalent functional relations which determine the same logarithmic ( $\ln x$ ) function. The function is said to be defined on E and to take its values in F or to be a function of an argument (or variable) running over E or, more simply, as a mapping of E into F.

- A.2.2 The mappings of a set E into a set F themselves constitute a new set, the set of mappings of E into F. In this view, an arbitrary element f of the set of mappings takes the value  $f(x)$  for an element x of E. Sometimes the notation  $f_x$  is employed. This is called index notation and the set E is then called the index set. The relation  $\langle y = f(x) \rangle$  is a functional relation in y which determines f. This is to be understood logically as naming the element y of F to be that element associated with x under the rule or process f.

The function determined by a relation of the form  $y = \langle x \rangle$  (or  $\langle x \rangle$ ) where  $\langle x \rangle$  is a combination of signs ultimately involving  $x$  is designated by the notation  $x \rightarrow \langle x \rangle$  (which in an abuse of language is sometimes abbreviated to  $\langle x \rangle$ ). For example, if  $X$  and  $Y$  are two generic subsets of a set  $E$ , the relation  $Y = \langle X \rangle$  is functional in  $Y$ ; the mapping of  $P(E)$  into  $P(E)$  determined is designated by  $X \rightarrow \langle x \rangle$ , or simply by  $\langle x \rangle$ . Often instead of saying  $\langle$  let  $f$  be a mapping of  $E$  into  $F \rangle$ , this is stated simply as  $\langle f: E \rightarrow F \rangle$ . Diagrams are often employed to describe situations involving several mappings. For instance,



designates that  $f$  maps  $A$  into  $B$  which in turn is mapped into  $C$  by  $g$  and into  $D$  by  $v$  while  $C$  is also mapped into  $D$  by  $h$ .

The relation of equality  $\langle f = g \rangle$  between mappings of  $E$  into  $F$  is equivalent to the relation  $\langle \forall x \in E, f(x) = g(x) \rangle$ . A function defined on a set  $E$  and taking the same value  $a$  for all elements  $x$  of  $E$  is called constant on  $E$ ; it is determined by the functional relation  $y = a$ . The mapping of  $E$  into  $E$  which associates each element  $x$  of  $E$  with itself is called the identity map. It is determined by the functional relation  $y = x$ . The mapping of an arbitrary subset  $A$  of  $E$  into  $E$  which associates with each element  $x \in A$  the element  $x$  itself considered as an element of  $E$  is called the canonical mapping of  $A$  into  $E$ . Given a mapping  $f$  of  $E$  into itself, the elements  $x \in E$  such that  $\langle f(x) = x \rangle$  are called invariant under  $f$ . This is extended by saying that  $x$  is invariant under a set of mapping when it is invariant under each of them.

A.2.3 Let  $f$  be a mapping of  $E$  into  $F$ , and  $X$  an arbitrary subset of  $E$ . The image of  $X$  under  $f$  is the subset  $Y$  of  $F$  consisting of those elements  $y$  that possess the property:

$$\langle \exists x \in E \ni (x \in X \text{ and } y = f(x)) \rangle$$

In this way, we determine a relation between the subsets  $X$  and  $Y$  which is functional in  $Y$ , and therefore determines a mapping of  $P(E)$  into  $P(F)$ . This mapping is called the extension of  $f$  to the set of subsets which, in an abuse of language, is again denoted by  $f$  and written as  $Y = f(X)$ . For all  $f$  and  $x$ , one has  $f(\emptyset) = \emptyset$  and  $f(\{x\}) = \{f(x)\}$ . Again in an abuse of language the value  $f(x)$  of  $f$  for  $x$  is also called the image of  $x$  under  $f$ . Sometimes when an arbitrary element  $y$  of  $F$  can be written as  $\langle y \in f(E) \rangle$ , we say  $y$  is of the form  $f(x)$ . Also speaking loosely, the image  $f(E)$  of  $E$  under  $f$  is called simply the image of  $f$ . When  $f(E) = F$ , or for arbitrary  $y \in F$  there always exists  $x \in E$  such that  $y = f(x)$ , then we say that  $f$  maps  $E$  onto  $F$ . Sometimes the onto map is also referred to as a surjection. Sometimes, particularly in geometric situations, instead of mapping we use the term transform, e.g.,  $f$  transforms  $x$  into  $f(x)$  and  $X$  into  $f(X)$ . In this terminology  $f(x)$  and  $f(X)$  are the transforms.

When  $f$  maps  $E$  into itself, the subsets  $X \subset E$  such that  $f(X) \subset X$  are called stable under  $f$ . The terminology is extended to a class of maps when the subset is stable under each map of the class.

The following propositions hold for a map  $f$  of  $E$  into  $F$  with arbitrary subsets  $X$  and  $Y$  of  $E$ .

- 1)  $X \subset Y$  implies  $f(X) \subset f(Y)$
- 2)  $X \neq \emptyset$  implies  $f(X) \neq \emptyset$
- 3)  $f(X \cup Y) = f(X) \cup f(Y)$
- 4)  $f(X \cap Y) \subset f(X) \cap f(Y)$

A.2.4 The question of "inverse" elements is of great importance. The subset  $X$  of  $E$  consisting of the elements  $x$  such that  $f(x) \in Y$  is called the reciprocal image of  $Y$  under  $f$ . This defines a relation between  $X$  and  $Y$  which is functional in  $X$  and therefore defines a mapping of  $P(F)$  into  $P(E)$  which is called the reciprocal extension of  $f$  to the set of subsets, written  $f^{-1}$ . Note this means that  $X = f^{-1}(Y)$ . A case of particular interest is the set  $\{y\}$  consisting of a single element where  $f^{-1}(\{y\})$  consists of all those elements  $x \in E$  such that  $f(x) = y$ . The relations  $\langle f(x) = y \rangle$  and  $\langle x \in f^{-1}(\{y\}) \rangle$  are equivalent. Sometimes  $f^{-1}(y)$  appears instead of the more formal  $f^{-1}(\{y\})$ . The trace  $X_A$  of a subset  $X$  of  $E$  on a fixed subset  $A$  is just the reciprocal image of  $X$  under the canonical mapping of  $A$  into  $E$ . The following propositions hold for a map  $f$  of  $E$  into  $F$  with arbitrary subsets  $X$  and  $Y$  of  $E$ .

- 1)  $X \subset Y \Rightarrow f^{-1}(X) \subset f^{-1}(Y)$
- 2)  $f^{-1}(X \cup Y) = f^{-1}(X) \cup f^{-1}(Y)$
- 3)  $f^{-1}(X \cap Y) = f^{-1}(X) \cap f^{-1}(Y)$
- 4)  $f^{-1}(f(X)) \supseteq X$

It should be noted that the relation on intersections  $\langle f^{-1}(X \cap Y) = f^{-1}(X) \cap f^{-1}(Y) \rangle$  does not hold for any map of  $F$  into  $E$ . As stated in A.2.3 (4), in general only the weaker statement  $\langle f(X \cap Y) \subset f(X) \cap f(Y) \rangle$  is true. Likewise, the relation  $\langle f^{-1}(f(X)) = X \rangle$  has no analogue for an arbitrary extension of a map to the set of subsets. It is also possible for  $f^{-1}(x) = \emptyset$  for a non-empty  $X$  of  $F$ . It is necessary and sufficient that  $f$  be a mapping of  $E$  onto  $F$  for  $X \neq \emptyset$  to imply that  $f^{-1}(x) \neq \emptyset$ .

When a map  $f$  of  $E$  into  $F$  has the property that for all  $y \in F$  there exists at most one element  $x \in E$  such that  $y = f(x)$  (e.g. the set  $f^{-1}(\{y\})$  is empty or consists of a single element) then  $f$  is a bi-unique mapping of  $E$  into  $F$  or an injective map or simply an injection. In this case, for arbitrary subsets  $X$  and  $Y$  of  $E$ , the relation  $\langle f(X \cap Y) = f(X) \cap f(Y) \rangle$  does hold.

In ordinary usage the particular instance where the stronger condition that  $f^{-1}(\{y\})$  consists of one and only one element holds is of paramount importance. A map with this property is called a bi-unique mapping of  $E$  onto  $F$  or a bijection map or simply a bijection. Formally, given a map  $f$  of  $E$  into  $F$  such that for all  $y \in F$  there exists one and only one  $x \in E$  such that  $y = f(x)$  then  $f$  is a bijection. We also say that  $f$  is one-to-one and onto and in this case there is an inverse function without recourse to the extension to the set of subsets. When  $f$  is a bi-unique map of  $E$  onto  $F$ , the relation  $\langle y = f(x) \rangle$  is not only functional in  $y$ , but it is also functional in  $x$ . As a consequence of being functional in  $x$ , the function  $f$  determines a bi-unique map of  $F$  onto  $E$  which is called the inverse (or reciprocal) map of  $f$ . In this case, note that the reciprocal extension of  $f$  to the set of subsets  $f^{-1}$  is identical to the inverse function. A bijection can also be characterized as being at the same time a map of  $E$  onto  $F$  and a bi-unique map or injection of  $E$  into  $F$ .

A.2.5 Let  $g$  be the inverse function of  $f$ ; the relations  $\langle y = f(x) \rangle$  and  $\langle x = g(y) \rangle$  are equivalent. The inverse function of  $g$  is  $f$ . With  $f$  a bi-unique map of  $E$  onto  $F$ , the relation  $f(X) = \{f(X)\}$  holds for all  $X$  of  $E$ . Further, the extension of  $f$  is a bi-unique mapping of  $P(E)$  onto  $P(F)$ . A bi-unique mapping of  $E$  onto  $F$  and its inverse map realize a bi-unique correspondence between  $E$  and  $F$ . We say  $E$  and  $F$  are put into bi-unique correspondence by these maps. A bi-unique mapping of a set  $E$  onto itself is called a permutation of  $E$ ; the identity map is included and is also a permutation. When a permutation is identical to its inverse map, it is called involution. An example of this is  $X \rightarrow (X)$  of  $P(E)$  onto itself.

A.2.6 In the following propositions  $X$  is an arbitrary subset of  $E$  and  $Y$  an arbitrary subset of  $F$ , while  $f$  is a mapping of  $E$  into  $F$ .

- 1)  $f^{-1}(Y) = f^{-1}(Y \cap f(E))$
- 2)  $X \subset f^{-1}(f(X))$
- 3)  $f(f^{-1}(Y)) \subset Y$

The properties  $\langle$  for all  $Y$ ,  $f(f^{-1}(Y)) = Y \rangle$  and  $\langle$   $f$  is mapping of  $E$  onto  $F \rangle$  are equivalent. The properties  $\langle$  for all  $X$ ,  $f^{-1}(f(X)) = X \rangle$  and  $\langle$   $f$  is a bi-unique map of  $E$  into  $F \rangle$  are equivalent. The properties  $\langle$  for all  $X, Y$ ,  $f^{-1}(f(X)) = X$  and  $f(f^{-1}(Y)) = Y \rangle$  and  $\langle$   $f$  is a bi-unique mapping of  $E$  onto  $F \rangle$  are equivalent.

A.2.7 Given three sets  $E, F, G$  which may not be distinct, and a mapping  $f$  of  $E$  into  $F$  and a mapping  $g$  of  $F$  into  $G$ , then the mapping of  $E$  into  $G$  defined for  $x \in E$  by  $g(f(x))$  is called the composition map of  $g$  and  $f$ . This map is written as  $g \circ f$  or simply as  $gf$  when there is no ambiguity. The equality  $h = g \circ f$  is called a factorization of  $h$ . The order of composition is important. The reverse composition  $f \circ g$  makes no sense when  $G$  is distinct from  $E$ . When the three sets  $E, F, G$  are all identical then both compositions  $g \circ f$  and  $f \circ g$  are well defined, but in general they are not the same. Compositions maps are also extended to the set of subsets and with  $\varphi$  the composition of  $g$  and  $f$ ,  $X$  an arbitrary subset of  $E$  and  $Z$  an arbitrary subset of  $F$ :

- 1)  $\varphi(X) = g(f(X))$
- 2)  $\varphi^{-1}(Z) = f^{-1}(g^{-1}(Z))$

Composition maps inherit the property of being bijections (1-1 and onto). More precisely, if  $f$  is a (1-1, onto) map of  $E$  onto  $F$  and  $g$  is a (1-1, onto) map of  $F$  onto  $G$ , then  $g \circ f$  is a (1-1, onto) map of  $E$  onto  $G$ . (Compositions can be

extended to more than two functions and are associative. For instance given an additional map  $h$  of  $G$  into  $H$ , then  $h \circ (g \circ f) = (h \circ g) \circ f$  is a map of  $E$  into  $H$ . If  $f$  is a map of  $E$  into itself, the iteration of  $f$ , written  $f^n$  ( $n$  integer  $\geq 1$ ), is defined recursively as  $f^1 = f$  and  $f^n = f^{n-1} \circ f$ . The map  $f^n$  is the  $n^{\text{th}}$  iteration of  $f$  and the law of exponents holds, e.g.  $f^{m+n} = f^m \circ f^n$ .

A note of caution: the composition maps  $(f^{-1} \circ f)$  or  $(f \circ f^{-1})$  respectively of the reciprocal extension and the extension maps are not in general the identity maps of  $P(E)$  or  $P(F)$  respectively onto themselves. When  $f$  is a bijection, however, the above compositions do indeed give the identity maps. Conversely, if  $f$  maps  $E$  into  $F$ , and  $g$  maps  $F$  into  $E$  and, in addition,  $g \circ f$  is a permutation of  $E$  and  $f \circ g$  is a permutation of  $F$  then  $f$  is a bi-unique mapping of  $E$  onto  $F$  and  $g$  is a bi-unique mapping of  $F$  onto  $E$ . If, further,  $g \circ f$  is an identity map of  $E$  onto  $E$ , then  $g$  is the inverse map of  $f$ .

A.2.8 Let  $f$  be a map of  $E$  into  $F$ , and  $A$  an arbitrary subset of  $E$ ; the map  $f_A$  of  $A$  into  $F$  which, for arbitrary  $x \in A$ , takes the values  $f(x)$  is called the restriction of  $f$  to the subset  $A$ . The map can also be viewed as simply the composition of  $f$  and the canonical map of  $A$  into  $E$ . If two mappings  $f, g$  of  $E$  into  $F$  have the same restriction to  $A$ , then we say that they coincide in  $A$ . Conversely, we sometimes say that  $f$  is an extension of  $f_A$  to  $E$ .

A.2.9 A mapping of a set  $E$  onto a set  $F$  is also called a parametric representation of  $F$  by means of  $E$ . The set  $E$  is called the set of parameters and its elements are called simply parameters.

A family of elements of a set  $F$  is, by definition, a subset of  $F$  equipped with a parametric representation. In said another way, the correspondents under some mapping of some subset of  $E$ . The image of  $E$  under this mapping is the set of elements of the family. Note that two distinct families of elements of  $F$  can have the same subset of  $F$  as the set of their elements. The distinction is in the parametrization.

Any subset  $A$  of  $F$  can constitute a family of elements inasmuch as it would be sufficient to consider the family defined by the canonical map of  $A$  into  $F$ .

A family of elements of  $F$  defined by a mapping  $i \rightarrow x_i$ , of a set  $I$  into  $F$  is written as  $(x_i)_{i \in I}$  (or simply  $(x_i)$  when the index set is clear). If  $J$  is a subset of  $I$ , the family  $(x_i)_{i \in J}$  is called a sub-family of the family  $(x_i)_{i \in I}$  corresponding to  $J$ . It is defined by the restriction to  $J$  of the mapping  $i \rightarrow x_i$ .

REFERENCES

1. G.W. Graves and R. Thrall, "The Capacitated Transportation Problem with Convex Polygonal Costs", RM-4941-PR, The RAND Corporation, April 1966.
2. G.W. Graves and A. Whinston, "An Algorithm for the Quadratic Assignment Problem", Management Science, 16, 7, March 1970. This paper has also been published in the book, "Integer and Nonlinear Programming", edited by J. Abadie, published by North-Holland Publishing Company, 1970.
3. G.W. Graves and A. Whinston, "A New Approach to Discrete Mathematical Programming", Management Science, 15, 3, November 1968.

## OPERATING INSTRUCTIONS FOR MMSSSO COMPUTER CODE

The description of the parameters and data for the Mathematical Model of Supply Support for Space Operations will be illustrated by the following Examples 1-4 of actual computer output and by the following simple example. Assume we have five modules, and three shuttle craft of which the first craft is of a different type than the remaining two craft. The modules have the following physical characteristics:

TYPE	WEIGHT	DIAMETER	LENGTH
S-1	16.130	5.	28.
S-1	16.130	5.	28.
T-1	15.93	6.	6.
T-1	15.93	6.	6.
X-1	35.036	10.	10.

In order to fulfill the objectives of the program, the five modules must be delivered in specified time periods as follows:

MODULE	EARLIEST TIME	LATEST TIME
S-1	0.	12.
S-1	13.	24.
T-1	0.	12.
T-1	13.	24.
X-1	0.	12.

Due to the large length, the first two S-1 modules can not go on the first craft and due to its high weight the fifth X-1 modules must go on the first craft. The basic input data is illustrated by Example 1.

### INPUT

#### First Card

The first data card contains a set of general parameters. It is read in with a Fortran Namelist in the "MAIN" routine. The

namelist statement is:

NAMELIST/PARM/NPL,NST,NPT,LV1,LV2,GAP,FAC,P1,P2,P3,IPRT,ICN

The meaning of the parameters is as follows:

NPL: Number of modules or payloads. There must be exactly NPL module description cards included in the run.

Example NPL=5

NST: Number of shuttle craft or flights to be scheduled.

Example NST=3

NPT: Number of priority classes. Selected by the user to cut down on computation by limiting the higher order interchanges considered in the search procedure to one priority class at a time. In general not more than 30 modules should be assigned to the same priority class. Similar but not identical modules should be assigned to the same class. The quality of the final answer is not too sensitive to the number of priority classes.

Example NPT=1

LV1: Level of interchanges searched when the modules are free to move to any other shuttle craft. In general, this parameter should be left at LV1=1 or the computation time will be too high. This parameter has a default value of LV1=1.

Example LV1=1

LV2: Level of interchanges searched when the modules are constrained to interchange between pairs of shuttle craft. This limitation permits deeper searches without excessive computation time. In general, LV2=2 is very satisfactory. This parameter has a default value of LV2=2.

Example LV2=2

GAP: The permissible time interval for each shuttle craft must be greater than or equal to GAP. The length of the time interval for a shuttle craft is determined by the inter-

section of the individual time intervals of the modules assigned to the shuttle craft. This parameter has a default value of GAP=0.

Example GAP=0

FAC: The number of parametric steps to shift from complete emphasis on time interval feasibility to complete emphasis on weight-diameter-length balancing. With a good starting assignment FAC=0, in general FAC should be between 3-6. Some smoothing between time incompatibilities and physical loading characteristics as provided by FAC is essential for satisfactory performance. This parameter has a default value of FAC=3.

Example FAC=3

P1: The relative emphasis to be placed on weight balancing as opposed to diameter and length balancing. In general  $P1+P2+P3=1$  and  $P1 \geq 0, P2 \geq 0, P3 \geq 0$ . This parameter has a default value of P1=1.

Example P1=.2

P2: The relative emphasis to be placed on diameter balancing as opposed to weight and length balancing. This parameter has a default value of P2=0.

Example P2=.6

P3: The relative emphasis to be placed on length balancing as opposed to weight and diameter balancing. This parameter has a default value of P3=0.

Example P3=.2

IPRT: Intermediate printout of state of the program information is generated from the subroutine PRINTE by this parameter. When IPRT=0 no intermediate printout is generated, but when IPRT=1 the intermediate printout is generated. This parameter has a default value of IPRT=0.

Example IPRT=0

ICN: Indication to continue to next problem in stacked runs. The value ICN=0 causes termination while the value ICN=1 causes the program to read in a succeeding problem. This parameter has a default value of ICN=0.

Example ICN=0

N: Flag to permit stacked cases without updating payload data. The value N=0 allows the program to run stacked cases using payload data carried over from the previous case. The value N=1 allows payload data definitions to be input.

In summary, the parameter card for the example would read:

Col. 1 Col. 2

& PARM NPL=5,NST=3,NPT=1,P1=.2,P2=.4,P3=.6&END

Succeeding Cards

A card must be prepared for each module to be included in the run. The number of module cards must agree with the number specified by the parameter NPL. The input card for each module contains a four letter alpha-numeric descriptor, the weight, length and diameter of the module; the earliest and latest times the module can be scheduled and the first and last vehical possible for the module; a priority class designation and an initial assignment to a vehicle. These quantities are read with a Fortran FORMAT (A4,2X,5F6.0,4I2) external statement 11 in the "MAIN" routine. In terms of the example these cards would be punched as follows:

Columns

<u>1-4</u>	<u>5-10</u>	<u>11-16</u>	<u>17-22</u>	<u>23-28</u>	<u>29-34</u>	<u>35-36</u>	<u>37-38</u>	<u>39-40</u>	<u>41-42</u>
S-1	16.130	5.0	28.0	0.0	12.0	2	3	1	2
S-1	16.130	5.0	28.0	13.0	24.0	2	3	1	3
T-1	15.93	6.0	6.0	0.0	12.0	1	3	1	2
T-1	15.93	6.0	6.0	13.0	24.0	1	3	1	3
X-1	35.036	10.0	10.0	0.0	12.0	1	1	0	1

preferable to attack the problem of the relationship between the modules assigned to the same load directly. This would require abandoning linear models in favor of non-linear models.

The most widely studied non-linear integer programming problem is the "Quadratic Assignment" problem. [See reference 2]. In this model, there is an interaction factor between the objects to be assigned and a weighting or screening multiplier associated with the sites to which the objects are assigned. In its classical form, the objects are assigned uniquely to the sites.

#### Model III Quadratic Assignment Problem

Let,

$$I_{ip} = \begin{cases} 1 & \text{if module } i \text{ is assigned to site } p \\ 0 & \text{otherwise} \end{cases}$$

$$k_{ip} = \text{cost of assigning module } i \text{ to site } p$$

$$q_{ij} = \text{interference (or compatibility) of modules } i \text{ and } j$$

$$c_{pq} = \text{weighting multiplier between sites}$$

then subject to

$$\sum_p I_{ip} = 1 \quad \text{for each } i \text{ (each module must be assigned)}$$

$$\sum_i I_{ip} \leq 1 \quad \text{for each } p \text{ (at most one to a site)}$$

$$\min \sum_i \sum_p k_{ip} I_{ip} + \sum_{i,j} \sum_{p,q} q_{ij} c_{pq} I_{ip} \cdot I_{jq}$$

This model introduces directly a relationship between the modules being assigned. The chief difficulty is the requirement that only a single module is assigned to a site. This is easily extended and the interference can also be restricted to modules

assigned to the same site by taking  $c_{pq} = \begin{cases} 1 & \text{if } p=q \\ 0 & \text{otherwise} \end{cases}$ . This leads to the current model employed in the space supply support program.

#### Model IV Space Supply Support Problem

Let,

$$I_{ip} = \begin{cases} 1 & \text{if module } i \text{ is assigned to load } p \\ 0 & \text{otherwise} \end{cases}$$

$q_{ij}$  = interference or incompatibility of modules  $i$  and  $j$

then subject to

$$\sum_p I_{ip} = 1 \quad \text{for each } i \text{ (each module must be assigned)}$$

$$\min \sum_p \sum_{i>j} q_{ij} I_{ip} \cdot I_{jp}$$

In this simplified version of the quadratic assignment problem any number of modules can be assigned to the same load and the special form of the weighting factors  $c_{pq}$  allow then to be absorbed into a simplification of the extremal function. The costs  $k_{ip}$  of assigning a module to a load seem irrelevant and have been dropped in the current formulation. However, they might be utilized to control the number of loads utilized in preference to the current strategy of opening an additional load only when the solutions to a problem prove undesirable. The interference factors  $q_{ij}$  are currently pairwise factors thereby keeping the problem quadratic. These  $q_{ij}$  factors can be used to reflect both scheduling time and general loading compatibility. To reflect scheduling time, the time intervals  $T_j = [\text{earliest time}, \text{latest time}]$  are defined and,

$$q_{ij} = \infty \text{ if } T_i \cap T_j = \emptyset$$

In general, if a compatible time for scheduling exists, the interference factor is taken as,

$$q_{ij} = p_1 \cdot w_i \cdot w_j + p_2 \cdot d_i \cdot d_j + p_3 \cdot l_i \cdot l_j$$

where  $w_i$  is the weight of module  $i$

$d_i$  is the diameter of module  $i$

$l_i$  is the length of module  $i$

and the  $p_1$ ,  $p_2$  and  $p_3$  are parametric weighting factors  $p_1 \geq 0$ ,  $p_2 \geq 0$ , and  $p_3 \geq 0$  and  $p_1 + p_2 + p_3 = 1$ . These parametric factors are used to reflect the degree of difficulty these physical characteristics might occasion where modules are associated in the same load. The physical dimensions are general loading characteristics and it might be much more effective to simply construct a table of  $q_{ij}$  reflecting loading incompatibility by module types. If this more flexible procedure is followed, the model could be extended to include triple and perhaps quadruple interference factors between module types. The present scheme of employing general physical loading characteristics to determine loading incompatibility is most appropriate when large numbers of modules are to be assigned to the same load. Undoubtedly much stronger results would accrue to the more flexible tabular scheme when at most three or four modules would be assigned to the same load.

Having examined models and problem formulation, the next step is to investigate solution technique. Problems in this area of interest here can be approached in at least two

basic ways. The first way is to formulate the model in terms of somewhat artificially introduced variables such as the  $I_{ij}$  above and then to impose large numbers of constraints such as  $\sum_p I_{ip} = 1$  in order to make the variables behave appropriately. In this approach, the solution is defined implicitly as those variables which satisfy the constraints. This approach has been extensively cultivated and in areas where the problem is naturally formulated in terms of continuous variables, has led to excellent results. This is the traditional domain of mathematical programming. The attempt, however, to extend these results and tools into areas where the introduction of the variables is artificial and the number of constraints large has been much less successful. This is particularly true when, as here, the variables cannot be treated as continuous variables but must be forced to assume integer values. The best known solution technique in this line of development is the "Gomory Cut" for solving linear integer programming problems. However as pointed out above, the linear models are incapable of dealing with the critical problem of the interaction of modules assigned to the same vehical. This technique has also proved quite ineffective with large problems.

The second fundamental approach is to generate and enumerate the desired assignments directly and explicitly. This approach seems most appropriate when the problem is naturally discrete and the space easily generated by combinatorial techniques as is the case here. Technically, our problem is to generate and enumerate the space of all ways of the set

- IX2: at value IX2=0 no gain has yet been achieved in the current priority class while at value IX2=1 a gain has been recorded
- ID: number of modules moved in current trial of interchanges
- LT: module-vehical correspondents to be read from MAPR (J(LT)),MAPC(J(LT)) which are presently assigned
- LS: module-vehicle correspondents to be read from MAPR (J(LS)),MAPC(J(LS)) which is to replace the pair specified by LT
- LEVEL: the maximum number of interchanges to be considered in the current sequence of trial interchanges

The next line of output is the label "J,V VECTORS".

This is followed by the vectors printed with `FORMAT(1H□,I6,9I12)` and `FORMAT(1H□,10F12.6)` respectively:

- J(IC): a linear numbering of the possible assignment pairs specified by MAPR and MAPC
- V(IC): the change in the interference function to be achieved by switching the assignment pair given by corresponding J(IC) with the current one

These vectors are followed by the vector printed with `FORMAT('TL=',10F12.6)`

- TL(IT): value of V which has been reached at the IT<sup>th</sup> movement when searching up to LEVEL.

There are two output statements in the program calling "PRINTB". The first is at the end of subroutine "LPCEN" after generation of an initial assignment. The second is near the end of "CUTDAR" and is used after every transformation of assignments.

EXAMPLE 1  
(INPUT-DATA)

NPL= 67,NST= 32,NPT= 2,LV1= 1,LV2= 2,GAP=0.0 ,FAC=3.0000000 ,P1= 0.0 ,P2=0.5000000 ,P3=0.5000000 ,IPRT= 0,ICN= 0

END

NO.	MODULE	WEIGHT	DIAMETER	LENGTH	TF	TF	LE	LF	NC	IV
1	1	1.61	2.80	0.50	0.0	12.0	1	3	1	1
2	1	1.61	2.80	0.50	13.0	24.0	16	18	1	14
3	2	1.52	2.80	0.60	0.0	12.0	5	15	1	5
4	2	1.59	2.80	0.60	13.0	24.0	20	32	1	20
5	3	3.50	3.60	1.00	0.0	12.0	5	15	1	5
6	3	3.50	3.60	1.00	13.0	24.0	20	32	1	20
7	4	0.15	0.70	0.50	0.0	12.0	1	3	1	1
8	13	2.50	6.00	1.50	0.0	12.0	5	15	1	5
9	16	1.50	3.20	1.00	0.0	12.0	5	15	1	5
10	17	3.54	4.20	1.00	0.0	12.0	5	15	1	5
11	19	1.50	1.50	1.00	13.0	24.0	20	32	1	20
12	22	0.50	1.20	1.00	0.0	12.0	1	3	1	1
13	22	0.50	1.20	1.00	13.0	24.0	16	18	1	14
14	23	0.10	1.00	0.40	0.0	6.0	1	3	1	1
15	23	0.10	1.00	0.40	7.0	12.0	1	3	1	1
16	23	0.10	1.00	0.40	13.0	13.0	16	18	1	14
17	23	0.10	1.00	0.40	19.0	24.0	16	18	1	14
18	19	3.44	3.00	1.00	13.0	24.0	20	32	1	20
19	24	0.04	0.40	0.40	0.0	12.0	5	15	1	5
20	24	0.04	0.40	0.40	13.0	24.0	20	32	1	20
21	25	3.48	3.40	1.00	0.0	12.0	5	15	1	5
22	25	3.48	3.40	1.00	13.0	24.0	20	32	1	20
23	27	0.50	2.00	1.50	0.0	12.0	4	4	0	4
24	27	0.50	2.00	1.50	0.0	12.0	4	4	0	4
25	27	0.50	2.00	1.50	0.0	12.0	4	4	0	4
26	27	0.50	2.00	1.50	13.0	24.0	19	19	0	19
27	27	0.50	2.00	1.50	13.0	24.0	19	19	0	19
28	28	1.50	3.20	0.65	13.0	24.0	20	32	1	20
29	28	1.59	3.20	0.65	13.0	15.0	20	32	1	20
30	29	1.59	3.20	0.65	17.0	20.0	20	32	1	20
31	29	1.59	3.20	0.65	21.0	24.0	20	32	1	20
32	30	3.84	5.10	1.20	13.0	24.0	20	32	1	20
33	31	1.56	3.20	0.65	0.0	12.0	5	15	1	5

EXAMPLE 1 (Cont.)

34	32	3.64	4.50	1.20	13.0	24.0	20	32	1	20
35	33	1.55	3.20	0.65	0.0	6.0	5	15	1	5
36	33	1.55	3.20	0.65	7.0	12.0	5	15	1	5
37	33	1.55	3.20	0.65	13.0	13.0	20	32	2	20
38	23	1.55	3.20	0.65	19.0	24.0	20	32	2	20
39	34	0.16	1.10	0.50	13.0	24.0	16	19	2	16
40	35	0.25	1.20	1.00	0.0	12.0	5	15	2	5
41	35	0.25	1.20	1.00	13.0	24.0	20	32	2	20
42	36	0.04	0.40	0.20	0.0	12.0	1	3	2	1
43	37	0.06	1.30	0.40	0.0	12.0	1	3	2	1
44	38	0.10	1.70	0.35	0.0	4.0	1	3	2	1
45	38	0.10	1.70	0.35	5.0	8.0	1	3	2	1
46	38	0.10	1.70	0.35	9.0	12.0	1	3	2	1
47	38	0.10	1.70	0.35	13.0	16.0	16	19	2	16
48	38	0.10	1.70	0.35	17.0	20.0	15	18	2	16
49	38	0.10	1.70	0.35	21.0	24.0	15	18	2	16
50	39	1.59	3.20	0.65	13.0	19.0	20	32	2	20
51	39	1.59	3.20	0.65	19.0	24.0	20	32	2	20
52	40	1.59	3.20	0.65	0.0	12.0	5	15	2	5
53	42	1.53	3.20	0.50	13.0	13.0	20	32	2	20
54	42	1.53	3.20	0.50	19.0	24.0	20	32	2	20
55	42	1.59	3.20	0.65	0.0	4.0	5	15	2	5
56	43	1.59	3.20	0.65	5.0	8.0	5	15	2	5
57	43	1.59	3.20	0.65	9.0	12.0	5	15	2	5
58	44	3.64	4.50	1.20	13.0	24.0	20	32	2	20
59	46	0.10	1.70	0.35	13.0	24.0	16	18	2	16
60	47	0.32	1.00	0.50	0.0	3.0	1	3	2	1
61	47	0.32	1.00	0.50	4.0	6.0	1	3	2	1
62	47	0.32	1.00	0.50	7.0	9.0	1	3	2	1
63	47	0.32	1.00	0.50	10.0	12.0	1	3	2	1
64	48	0.27	1.80	1.20	0.0	12.0	1	3	2	1
65	49	0.27	1.80	1.20	13.0	24.0	16	18	2	16
66	49	0.15	1.10	0.50	13.0	24.0	16	18	2	16
67	50	3.50	4.20	1.00	0.0	12.0	5	15	2	5

EXAMPLE 2

(Normal Output)

0.640=0.5=0.2000  
 270502345 04

	MIN	MAX	WEIGHT	DIAMETER	LENGTH	MODULES				
1	7.0	8.0	2.533	6.700	2.350	1	12	45	62	
2	10.0	12.0	0.822	5.900	2.650	15	42	46	53	64
3	4.0	3.0	1.049	6.700	2.650	7	14	43	44	60 61
4	0.0	12.0	1.500	6.000	4.500	23	24	25		
5	0.0	12.0	3.154	6.000	1.250	3	33			
6	0.0	12.0	5.077	6.600	1.650	21	57			
7	5.0	9.0	3.186	6.400	1.650	9	56			
8	0.0	12.0	2.500	6.000	1.500	8				
9	0.0	6.0	1.553	3.200	0.650	35				
10	0.0	12.0	3.544	4.200	1.000	10				
11	7.0	12.0	1.553	3.200	0.650	36				
12	0.0	4.0	1.503	3.200	0.650	55				
13	0.0	12.0	3.500	4.200	1.000	67				
14	0.0	12.0	1.633	3.600	1.050	19	52			
15	0.0	12.0	2.754	4.800	2.000	5	40			
16	17.0	20.0	2.213	5.700	1.850	2	13	48		
17	13.0	15.0	0.622	5.500	2.450	16	39	47	65	
18	21.0	24.0	0.449	5.500	1.600	17	49	59	66	
19	13.0	24.0	1.000	4.000	3.000	26	27			
20	13.0	24.0	1.633	3.600	1.050	20	28			
21	13.0	19.0	1.783	4.400	1.500	41	53			
22	13.0	24.0	3.424	3.400	1.000	22				
23	17.0	20.0	5.037	6.200	1.650	18	30			
24	13.0	24.0	3.844	5.100	1.200	32				
25	13.0	24.0	2.644	4.500	1.200	34				
26	13.0	16.0	2.186	6.000	1.250	4	29			
27	13.0	24.0	3.644	4.500	1.200	58				
28	13.0	18.0	3.093	4.700	1.650	11	50			
29	21.0	24.0	3.146	6.400	1.300	31	38			
30	19.0	24.0	1.503	3.200	0.650	51				
31	19.0	24.0	5.037	6.300	1.500	6	54			
32	13.0	12.0	1.553	3.200	0.650	37				

EXAMPLE 3

(Normal Output After Switches)

IXI= 0 GAP=0. F=0.0000

G= 0.27934470E 04

LOAD	TMIN	TMAX	WEIGHT	DIAMETER	LENGTH	MODULES				
1	7.0	8.0	2.533	6.700	2.350	1	12	45	62	
2	10.0	12.0	0.829	5.900	2.650	15	42	46	63	64
3	4.0	3.0	1.049	6.700	2.650	7	14	43	44	60 61
4	0.0	12.0	1.500	6.000	4.500	23	24	25		
5	0.0	12.0	3.156	6.000	1.250	3	33			
6	0.0	12.0	2.484	7.400	1.000	21				
7	5.0	2.0	3.126	6.400	1.650	9	56			
8	0.0	12.0	2.500	6.000	1.500	8				
9	0.0	4.0	1.807	4.400	1.650	35	40			
10	0.0	12.0	3.544	4.200	1.000	10				
11	0.0	12.0	3.146	6.400	1.300	36	57			
12	0.0	4.0	1.503	3.200	0.650	65				
13	0.0	12.0	3.500	4.200	1.000	67				
14	0.0	12.0	1.532	3.600	1.050	19	52			
15	0.0	12.0	3.504	3.600	1.000	5				
16	17.0	20.0	2.213	5.700	1.850	2	13	49		
17	13.0	16.0	0.429	5.600	2.450	16	39	47	65	
18	21.0	24.0	0.469	5.500	1.600	17	49	59	66	
19	13.0	24.0	1.000	4.000	3.000	26	27			
20	12.0	24.0	1.433	3.600	1.950	20	28			
21	13.0	13.0	1.753	4.400	1.500	41	53			
22	13.0	24.0	3.484	3.600	1.000	22				
23	17.0	20.0	5.927	6.200	1.650	18	30			
24	13.0	24.0	3.244	5.100	1.200	22				
25	13.0	24.0	3.644	4.500	1.200	34				
26	13.0	16.0	3.136	6.000	1.250	4	29			
27	13.0	24.0	3.644	4.500	1.200	58				
28	13.0	18.0	3.022	4.700	1.650	11	50			
29	21.0	24.0	3.146	6.400	1.300	31	38			
30	19.0	24.0	3.126	6.400	1.150	51	54			
31	13.0	24.0	2.504	3.600	1.000	6				
32	13.0	12.0	1.553	3.200	0.650	37				

0.000000000000000000

1 12 12 15 15  
2 1 2 1 2  
3 3 4 3 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 29 30 31 32 33 34 35 36  
42 3 IX1= 1 IX2= 0 IO= 1 LT= 3 LS= 6 LEVEL= 3

J.V. VECTORS  
1 3 6 2 4 5  
-5.0000000 -3.274007 -2.800003 2.000003 0.800007 0.010007

FILE=\*\*\*\*\*

G= 0.27034470E 04 GH= 0.80009748E 00  
ASSIGNMENTS

1 1 12 12 15 15  
2 1 2 1 2  
3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 28 29 30 31 32 33 34 35 36  
42 3 IX1= 1 IX2= 0 IO= 1 LT= 2 LS= 5 LEVEL= 2

J.V. VECTORS  
1 4 6 2 3 5  
-2.000000 -4.064035 -3.600001 6.850001 -0.880007 -0.630000

FILE=\*\*\*\*\*

G= 0.27034470E 04 GH= 0.91009722E 00  
ASSIGNMENTS

1 1 12 12 15 15  
2 1 2 1 2  
3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 28 29 30 31 32 33 34 35 36  
42 3 IX1= 1 IX2= 0 IO= 1 LT= 3 LS= 6 LEVEL= 2

J.V. VECTORS  
1 3 5 2 4 5  
-7.000000 -4.776085 -3.819989 0.000003 -0.700009 -0.010007

FILE=\*\*\*\*\*

G= 0.27034470E 04 GH= 0.20009933E 01  
ASSIGNMENTS

1 1 12 12 15 15  
2 1 2 1 2  
3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 28 29 30 31 32 33 34 35 36  
42 3 IX1= 1 IX2= 0 IO= 1 LT= 1 LS= 4 LEVEL= 2

J.V. VECTORS  
1 3 6 1 4 5  
-6.000000 -2.000000 -4.300007 -2.000003 4.740005 -2.070003

FILE=\*\*\*\*\*