014524-9-T
SEL 136

METAPHOR (Version 1): User's Guide

D. G. FURCHTGOTT

Under the direction of
J. F. MEYER

July 1979

Prepared for

National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23365

G. E. Migneault, Technical Officer

NASA Grant NSG 1306

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

SYSTEMS ENGINEERING LABORATORY

THE UNIVERSITY OF MICHIGAN, ANN ARBOR

014524-9-T
SEL 136

METAPHOR (Version 1): User's Guide

By
D. G. Furchtgott

Under the Direction of
J. F. Meyer

Prepared For
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23365
G. E. Migneault, NASA Technical Officer
NASA Grant NSG 1306

June 1979

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

This report is a user's guide for the first version of *METAPHOR**, an interactive software package to facilitate performability modeling and evaluation. A companion "Programmer's Guide" for *METAPHOR* (Version 1) has already been published [1]. As the capability of *METAPHOR* is extended via incorporation of additional evaluation programs, revised or supplemented guides will be prepared in order to maintain an up-to-date documentation of the system. It is assumed that the reader is familiar with the context of *METAPHOR*, that is, the performability modeling and evaluation methods developed under the subject grant and described in a number of previous reports and publications [2]-[12].

As we currently envision *METAPHOR*, it is the prototype of a software package that, ultimately, will contain programmed tools to facilitate each step of performability model construction and model solution. In certain steps, such facilitation will take the form of complete automation; in other cases, particularly steps involving model construction, an interactive mode will be necessary wherein the programmed tool acts strictly as an aid. More specifically, the major steps to be facilitated are:

1) Construction of the base model,

**MICHIGAN EVALUATION AID FOR PERPHORMABILITY*

2) Elaboration of the base model into a model hierarchy,

3) Formulation of the capability function in terms of the interlevel translations between adjacent models of the hierarchy,

4) For each accomplishment level a, computation of the base model trajectory set U_a that corresponds to a,

5) For each trajectory set U_a, computation of its probability (the performability value for accomplishment level a).

In addition to facilitating specific steps of the modeling and evaluation process, *METAPHOR* is intended to serve as a performability evaluation tutor for a person who is learning to use its programs.

In developing Version 1 of *METAPHOR*, emphasis was placed on obtaining a general structure that can accommodate the various types of evaluation programs that are planned for the system. In addition, Version 1 contains specific programs which facilitate steps 1) and 5) outlined above. Finally, the tutorial aspect of *METAPHOR* is fairly well developed in Version 1 with an extensive repertoire of HELP requests, along with a preprogrammed series of questions relating to specific topics.

METAPHOR is written in *APL* [13]-[14], chosen because of its notational compactness and array handling abilities. However, the eventual translation of the prototype package into a faster and more portable language such as FORTRAN may be desired, and this report should also provide valuable documentation for such a process.

This report is organized as follows. Section 2. presents general information concerning *METAPHOR* and presents some hints on using *METAPHOR* effectively. Next, as illustrations of *METAPHOR*'s use, various example systems are studied and their performabilities calculated in Section 3. In Section 4. are given detailed descriptions of each presently implemented *METAPHOR* command, while, finally, Section 5. describes each available *METAPHOR* array generator.

Throughout this guide, we have striven to include as many clarifying examples as possible. Numerous complete *METAPHOR* sessions have been included so as to demonstrate as fully as possible the package and its use.

2. Basic Information

2.1. Terms and Definitions

As stated in the Introduction, this report presupposes some familiarity with performability modeling and evaluation techniques. In addition, a few terms will be used within this report to describe items with no counterpart in the theoretical development. In particular, a model parameter (or parameter) will refer to a characteristic of a performability model which the analyst can specify. For example, the number of phases in the model, the number of states in each of those phases, and the P matrices for each of those phases are all model parameters. That is, each piece of information which *METAPHOR* collects regarding the performability model is a parameter; parameters can be viewed and altered by the user at any time.

Also, the term time-invariant basic variable will be used to denote a Bernoulli basic variable whose distribution function is time invariant, and in particular, one which is statistically independent of any other basic variable. For example, in the case of aircraft computer performability models the basic variable "weather at the destination airport," (taking on values in the set {Category III, not Category III}) is Bernoulli, is usually statistically independent of any other basic variable, and so would be a time-invariant basic variable. The distinction is made in *METAPHOR* (though not in the theory, per se) for the convenience of the user. The probabilities of these variables need be stated only once, at the beginning of the evaluation.

Knowledge of APL, though useful, is certainly not necessary. The only APL-ish concept employed that is perhaps not commonly found in other programs is that of a vector, and especially, that of a vector used for inputting data. However, the idea is very close to that of a one dimensional array in FORTRAN or other "conventional" language and so should cause no problems.

2.2. Use of APL Vector and Scalar Notation in METAPHOR

Because inputting data to *METAPHOR* often requires using vectors, a brief review of *APL* vector (i.e., single dimensional array) and scalar notation is presented below for the convenience of the reader. However, this report is not intended to be a guide to *APL* and hence the discussion is brief. Multiple dimensioned arrays (e.g., matrices) will not be covered as these are not employed for data input.

Scalars are single numbers-- examples are:

```
0
3
47.15
7.667E4
+7
¯6.359
¯5.11459E53
+6.74995E¯8 .
```

Note that negation is indicated with a raised negative sign "¯", and that numbers in exponential form are written as a number followed by "*E*" followed by an integral power of 10.

Vectors are used in *METAPHOR* for inputting values which are similar in nature, as for example, the number of states in each phase of the model. A vector is entered as a row of numbers separated by blanks or commas. For instance,

```
3  5 7   9
0.314  .271,0.354,   .678
```

Here, for the first vector, the first number may represent the number of states in the first phase (3), the second number the number of states in the second phase (5), and so forth, while the second vector may represent the probability of the first time-invariant basic variable (0.314), the second the probability of the second time-invariant basic variable (0.271), and so on. In general, if we are presented a group of k objects (e.g., phases), the n^{th} entry of a vector represents some information concerning the n^{th} object. Furthermore, the vector must have length k, i.e., each object must be accounted for in the vector. Of course the ordering of the vector is of utmost importance: 3 5 7 9 is <u>not</u> the same as 9 5 3 7 or 5 9 7 3. Thus, we must be sure of the correspondence between elements of a vector and objects in a group. Also note that even 0's must be accounted for, e.g., 1 1 0 0 is not the same as 1 1 or 1 1 0 or 1 1 0 0 0. If a vector is extremely long, it can be entered in sections by using the quad symbol (⎕) as a place holder; the quad must be preceded by a comma. Thus,

```
⎕:  0   5 7   8.32   6.7   9  14 ,⎕
⎕:  9   4  34 4556,88 334,  303 4.22 4040,⎕
⎕:  35 44 455
```

would input the single 19 element vector:

0 5 7 8.32 6.7 9 14 9 4 34 4556 88 334 303 4.22 4040 35 44 455 .

If user defined variables (see Section 4.3.) are being employed, the catenation (,) function can be useful for inputting difficult to type or repetitious portions of vectors. As an illustration, consider the user defined variables

```
U4←1 1 1 1
U5←1 1 1 1 1
U6←1 1 1 1 1 1
U7←1 1 1 1 1 1 1
```

and suppose the following vectors must be input:

```
1 0 1 1 1 1 1
1 0 0 1 1 1 1
1 1 1 1 0 0 0
1 1 1 1 1 1 1
1 1 1 1 1 1 0
1 1 1 1 1 0 0   .
```

Then the user can type

```
⎕:  1 0 0,U5
⎕:  1 0 0 , U4
⎕:  U4,0 0 0
⎕:  U7
⎕:  U6,0
⎕:  U5,0 0
```

2.3. Entering Trajectory Sets

To evaluate a performability model, the probability of each trajectory set must be calculated. For *METAPHOR* to make these computations, however, these trajectory sets must first be somehow characterized and input.

In METAPHOR, trajectory sets are input phase by phase utilizing vectors as follows. Consider a trajectory set V over m phases. Then for a given phase k, enumerate the states $0,1,2,\ldots,n-1$ and associate a binary vector $a_{n-1}\ a_{n-2}\ \cdots\ a_0$ such that that for $0 \le i \le n-1$

$$a_i = \begin{cases} 1 & \text{if state } i \text{ lies within the trajectory set (for that phase)} \\ 0 & \text{otherwise.} \end{cases}$$

$$= \begin{cases} 1 & \text{if } q_i \in \xi m(v) \text{ (in the notation of [4])} \\ 0 & \text{otherwise.} \end{cases}$$

(See also the concept of a "degree of freedom" variable [15].) Then we input that vector for each phase of the trajectory set. For example, suppose we have a trajectory space

$$\{0,1,2,3\} \times \{0,1\} \times \{0,1,2,3\} \times \{0,1,2,3,4\}$$

i.e., a model having 4 phases with phases 1 and 3 having state space $\{0,1,2,3\}$, phase 2 with state space $\{0,1\}$, and phase 4 state space $\{0,1,2,3,4\}$. Then the trajectory set

[{0,1,2} 1 2 *]

would be represented by the sequence of vectors

0 1 1 1
1 0
0 1 0 0
1 1 1 1 1 .

Naturally, as noted in Section 2.2., the ordering of the vectors above is critical: 0 1 0 0 is <u>not</u> equivalent to 0 0 1 0.

Note that for phase k, k not the final phase,

$$a_i = g_{ii}(k),$$

where $G(k) = [g_{ij}(k)]$ $(1 \leq i \leq n,\ 1 \leq j \leq n)$ is the characteristic matrix for phase k (see [4], p.51). That is, the vector characterizing the trajectory set V within phase k $1 \leq k \leq m$ is nothing more than the diagonal of G(k). Thus for the above example

$$G(1) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$G(2) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$G(3) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$G(4) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Hence, when inputting a trajectory set, *METAPHOR* will ask for *"THE G DIAGONAL"* of the k^{th} phase. Note, too, that for the final phase, i.e., phase m,

$$a_i = f_i(m)$$

here $F(m) = [f_i(m)]$ ($1 \le i \le n$ = number of states in phase m) is the characteristic vector (see [4]), and so the last vector discussed in the above example is the transpose of the F vector. Hence, *METAPHOR* will request the F vector when inputting a trajectory set. (This is a slight abuse since the user is actually entering the transpose of F, i.e., F^T. However, entering F^T is easier than entering F, and asking for the F vector is less confusing than asking for the transpose of the F vector.)

The following *METAPHOR* session demonstrates how the above trajectory could be entered:

```
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  1  1  1
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0
PHASE 3:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  1  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  1  1
```

2.4. Inputting Information Associated with Trajectory Sets

Along with every trajectory set provided to *METAPHOR*, up to two other pieces of information may be requested. The first is an initial state distribution, the I vector. When this is requested, enter a probability vector such that its i^{th} entry is the probability that the system will initially be in state i. Of course, the entries in the I vector should sum to 1; if not, *METAPHOR* will print an error message and again ask for the I vector. The I vector is requested for each trajectory set--the same I vector should be used for all trajectory sets.

The second item associated with trajectory set input is a vector characterizing the time-invariant basic variables. (See Section 2.1.) This vector, called in *METAPHOR* the *CONSTANT BASIC VARIABLE VECTOR*, is requested only if time-invariant basic variables have been specified. The number and probabilities of any time-invariant basic variables are specified before any trajectory sets are input: *METAPHOR* requests "*THE NUMBER OF CONSTANT BASIC VARIABLES*," and then asks for the "*PROBABILITIES OF EACH CONSTANT VARIABLE*." This refers to the probability of the event associated with each basic variable. For example, suppose we have two time invariant basic variables: 1) Category III weather at the destination airport, with probability 0.016, and 2) Autoland equipment on the ground at the destination airport does not work, with probability 0.01. Then the following sequence would inform *METAPHOR* of these variables:

NUMBER OF CONSTANT BASIC VARIABLES?
⎕: 2

PROBABILITIES OF EACH CONSTANT VARIABLE?
⎕: 0.016 0.01

If v_i is the i^{th} entry of the constant basic variable vector, and b_i is the i^{th} time-invariant basic variable, then

$$v_i = \begin{cases} 2 & \text{if both values of } b_i \text{ lie within the trajectory set, (i.e., } b_i \text{ is a "don't care"),} \\ 1 & \text{if only the value of } b_i \text{ corresponding to the given probability (i.e., the probability of the event associated with the basic variable as specified above) lies within the trajectory set,} \\ 0 & \text{otherwise.} \end{cases}$$

For instance, suppose the trajectory set we are entering requires Category III weather but makes no restriction on whether the ground autoland equipment is functioning. Then

ENTER THE 2 *ELEMENT CONSTANT BASIC VARIABLE VECTOR:*
⎕: 1 2

2.5. Operation of METAPHOR

Version 1 of *METAPHOR* is written in *APL*. Hence, use of *METAPHOR* is initiated by running *APL* and loading a workspace containing the *METAPHOR* package. This procedure is implementation dependent and so will not be described here. Once the workspace is loaded, *METAPHOR* is run by typing the function name

METAPHOR

METAPHOR then responds with

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
            VERSION 1

TYPE HELP FOR ASSISTANCE
⎕:
```

METAPHOR is now ready to accept commands. A prompt symbol composed of a quad followed by a colon (⎕:) is printed whenever a response from the user is expected.

2.6. METAPHOR Modes

Depending on the type of response expected from the user, *METAPHOR* will be in one of the 5 response modes COMMAND, COMMAND/REPLY, YES/NO, CALC, or COM. Figure 1 lists these modes along with their corresponding prompt symbols and with possible responses the user can make. When in COMMAND mode, the user can issue any of the commands discussed in Section 4. Often, *METAPHOR* will ask a question requesting some parameter value, e.g., *NUMBER OF PHASES?*, and in such cases, the mode is COMMAND/REPLY. The user can either issue another command (for instance, *HELP* or *CALC*) or reply to the question. In this mode, user defined variables can be utilized to input numerical values (see Section 4.3.), while occasionally the user can employ commands which automatically or semi-automatically generate numerical arrays (see Section 5.). *METAPHOR* will also present questions which require a YES or NO answer and this mode is

MODE	PROMPT SYMBOL	POSSIBLE RESPONSE
COMMAND	⎕:	Commands
COMMAND/REPLY	⎕:	Commands, Numbers, Variables, Sometimes array types
YES/NO	⎕:	Words, Numbers
CALC	? ⎕:	Expressions, EXIT
COM	***	Words, Null line

Figure 1. METAPHOR modes

called YES/NO. Only words and numbers can be entered-- *METAPHOR* scans these for some indication of affirmation or negation, specifically either 1 or *Y* for YES, or 0 or *N* for NO. YES has precedence in case both are present, but NO is assumed if neither YES nor NO is indicated. The user can also place *METAPHOR* in a calculator (CALC) mode (see Section 4.3.), in which case any valid *APL* expression input is evaluated and the result printed. This mode is characterized by the prompt symbol

?
⎕:

and can be left by typing *EXIT*. Finally, to document a *METAPHOR* session, the comment (COM) mode (see Section 4.4.) can be entered. Any sequence of symbols can be typed, but none will be processed; to leave, one gives a carriage return with no preceding characters (a null line). The COM mode causes every line to begin with ***.

2.7. METAPHOR Commands

Presently 9 commands are recognized by *METAPHOR*. These have the syntax

COMMAND [*ON*|*OFF*]

where the modifiers *ON* and *OFF* are used with the commands *BRIEF* and *ECHO*. The other commands are *ALTER*, *CALC*, *COM*, *DATA*, *EVAL*, *EXIT*, and *HELP*. Complete descriptions of these are given in Section 4., while examples of their use are presented in Section 3. Any command can be issued at any time *METAPHOR* is in either COMMAND or COMMAND/REPLY mode. Also, when in COMMAND/REPLY mode, 4 types of array specification commands are available for use when inputting P or H matrices: *GIVEN*, *IDENTITY*, *NFAIL*, and *DEDFAIL*. These matrix types are described in Section 5.

If a command is given while *METAPHOR* is processing an earlier command, the first command will be stacked; its execution will be continued once the second command has been completed. Any number of commands can be stacked, subject only to constraints imposed by the host *APL* program. Array specifications are also capable of being stacked. An example of such stacking is given in Figure 2.

2.8. METAPHOR as a Performability Evaluation Tutor

METAPHOR was designed to fulfill three broad design criteria: 1) To serve as a tool for performability modeling and evaluation, 2) To demonstrate the feasibility of performing the calculations associated with performability evaluation, and 3) To provide a tutorial basis for communicating the techniques of performability modeling and evaluation. The latter goal is appreciably mature in Version 1. Since the initiation of our work, we have recognized the necessity of providing the means of communicating the purposes and methodologies of performability analysis. Although to a large extent publications, presentations at meetings, and personal contacts provide such transfers of information, we feel that the availability of a package with which an analyst can experiment and learn will add significantly to the potential use of performability. Thus, *METAPHOR* was designed from its first stages as more than a

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
□ :   COM
*** THIS SESSION DEMONSTRATES THE STACKING
*** CAPABILITY OF METAPHOR COMMANDS
***
□:  EVAL

NUMBER OF PHASES?
□:  2

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  2  2


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  IDENTITY

PHASE 2:

WHAT TYPE OF P MATRIX?
□:  IDENTITY


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1¯2:

WHAT TYPE OF H MATRIX?
□:  COM
*** FIRST, WEΔLL STACK THE EVAL COMMAND WITH A DATA COMMAND
***
WHAT TYPE OF H MATRIX?
□:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
                        X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF


THE P MATRICES ARE:

 1 0
```

Figure 2. METAPHOR session illustrating the stacking of commands

```
 0 1

 1 0
 0 1
WHAT TYPE OF H MATRIX?
⎕:  COM
*** THE DATA COMMAND HAS BEEN EXECUTED--
*** WE'RE BACK TO THE EVAL COMMAND
***  NOW WE'LL STACK THE EVAL COMMAND WITH AN ALTER COMMAND
***
WHAT TYPE OF H MATRIX?
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS
X
ALTERING P


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  COM
*** NOW WE WILL STACK THE FIRST ALTER COMMAND WITH A SECOND ONE
***
WHAT TYPE OF P MATRIX?
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS
X
ALTERING P


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  COM
***  FINALLY, WE WILL STACK BOTH OF THE PREVIOUS ALTERS WITH A THIRD ONE
***
WHAT TYPE OF P MATRIX?
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS
X
ALTERING P


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME
```

Figure 2. METAPHOR session illustrating the stacking of commands -- continued)

```
PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  COM
***  LET'S TAKE A LOOK AT THE PRESENT P MATRICES
***
WHAT TYPE OF P MATRIX?
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
                        X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF


THE P MATRICES ARE:

 1 0
 0 1

 1 0
 0 1
WHAT TYPE OF P MATRIX?
⎕:  GIVEN
ENTER THE MATRIX, 1 ROW AT A TIME

ROW 1:
⎕:  0.01  0.99
ROW 2:
⎕:  0.99  0.01

PHASE 2:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  1E¯6
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  1
WHAT TYPE OF P MATRIX?
⎕:  COM
*** WE HAVE COVERED THE TOP ALTER OF THE STACK.   CHECK THE VALUE OF P
***
WHAT TYPE OF P MATRIX?
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
                        X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF
```

Figure 2. METAPHOR session illustrating the stacking of commands -- continued)

```
THE P MATRICES ARE:

  1.000000000E¯2    9.900000000E¯1
  9.900000000E¯1    1.000000000E¯2

  9.999900000E¯1    9.999950000E¯6
  0.000000000E0     1.000000000E0
WHAT TYPE OF P MATRIX?
⎕:  DEDFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001

PHASE 2:

WHAT TYPE OF P MATRIX?
⎕:  GIVEN
ENTER THE MATRIX, 1 ROW AT A TIME

ROW 1:
⎕:  0  1
ROW 2:
⎕:  1  0
WHAT TYPE OF P MATRIX?
⎕:  COM
***   WE HAVE NOW COVERED THE TOP TWO STAKED ALTER COMMANDS.
*** LET'S SEE WHAT THE CURRENT VALUE OF P IS
WHAT TYPE OF P MATRIX?
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
                        X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF


THE P MATRICES ARE:

 0.9990004998      0.0009995001666
 0                 1

 0                 1
 1                 0
WHAT TYPE OF P MATRIX?
⎕:  IDENTITY

PHASE 2:

WHAT TYPE OF P MATRIX?
⎕:  IDENTITY
```

Figure 2. METAPHOR session illustrating the stacking of commands -- continued)

```
WHAT TYPE OF H MATRIX?
⎕:  COM
*** AT LAST WE HAVE COVERED ALL THE ALTER COMMANDS.
*** MAKE SURE WE HAVE THE LAST MATRIX WE SPECIFIED
***
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
                        X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF


THE P MATRICES ARE:

     1               0
     0               1

     1               0
     0               1

WHAT TYPE OF H MATRIX?
⎕:  COM
*** NEXT, WE STACK THE PRESENT EVAL COMMAND WITH ANOTHER EVAL COMMAND
***
WHAT TYPE OF H MATRIX?
⎕:  EVAL

NUMBER OF PHASES?
⎕:  COM
*** FINALLY, WE STACK THE TWO EVAL COMMANDS WITH
```

Figure 2. METAPHOR session illustrating the stacking of commands -- continued)

```
*** ENTIRE METAPHOR PROGRAM
***
□:  METAPHOR

MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1

TYPE HELP FOR ASSISTANCE
□:
```

Figure 2. METAPHOR session illustrating the stacking of commands -- continued)

bookkeeping and calculator program. With it, an analyst can receive information concerning his possible next actions or further information concerning some question *METAPHOR* has asked. (See Section 4.9.) *METAPHOR* will guide the analyst through a performability model evaluation by collecting information in a logical sequence, hence insuring the inclusion of all steps. Finally, different types of consistency checks are made throughout the package to lessen the chance of error on the part of the analyst. First, some redundant information is requested, as, for example, when *METAPHOR* initially requests the number of phases and then asks for the number of states in each of those phases. Although the package could have been designed to request only the number of states in each phase (the number of phases would then be implicitly defined to be the number of values input), this was not done to prevent possible errors such as omitting the number of states in one phase. Secondly, consistency checks are made regarding the legality of all inputs. For example, all probabilities must be between 0 and 1, and all probability vectors must sum to 1. Also, phase lengths, failure rates, etc., must be positive, while quantities such as the number of phases and the number of states per phase must be positive integers. A third type of consistency check occurs in certain of the array generators such as *DEDFAIL* (see Section 5.1.) and *NFAIL* (see Section 5.4.). These tests are made on the reasonableness of component failures. If the failure rates are not within the range 10^{-1} to 10^{-10}, the user is asked to confirm the failure rate. Finally, a check is made of the performability result; in case sum of the the performabilities of all the accomplishment levels does not sum to one, a warning is printed.

2.9. Use of Command Files in METAPHOR

Two commands have been implemented in *METAPHOR* to support the convenient use of command files. (A command file is a file or device containing a sequence of commands to be used as input. The file is then "sourced," i.e., the program is directed to process commands from the file rather than from the terminal.) This technique allows the preservation of rather long or tedious *METAPHOR* sessions and so simplifies the modification of evaluations. [In MTS*, an *APL* file (say MY.COMMAND.FILE) is sourced with the command

)SOURCE MY.COMMAND.FILE

Some *APL* systems may not support command files.]

The two commands assisting with such command files are *BRIEF* (see Section 4.2.) and *ECHO* (see Section 4.6.). The command *BRIEF ON* prevents *METAPHOR* from printing any questions or remarks other than prompts for answers (with the exception that the performability result is always printed; hence using *BRIEF ON* in conjunction with a command file results in a *METAPHOR* session with very little output. For instance,

---

*Michigan Terminal System

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1

TYPE HELP FOR ASSISTANCE
⎕:  BRIEF ON
BRIEF BRIEF ON
⎕:  )SOURCE COMMANDFILE
PERFORMABILITY FOR THIS MISSION ← 0.00999997005  0.99000003
```

The second command, *ECHO ON*, causes *METAPHOR* to echo each input line. This is useful for obtaining a record of the input from a command file (since normally the input from a command file is not printed on the terminal). Thus, without *ECHO ON*,

```
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  0
```

while with *ECHO ON*,

```
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0
⎕:  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  UFINALVECTOR
⎕:  1  1  0  0
```

3. Examples of the Use of METAPHOR

To illustrate the use of METAPHOR, consider the following examples of performability evaluations carried out with the aid of METAPHOR. In most cases, the examples are extremely simple, and an analytical solution is given (though the derivation is not shown); these analytical solutions are presented as checks. The reader will note that numerous examples have been provided--in particular, each capability of METAPHOR has been illustrated with its own example. Hopefully, the volumne of so many examples will neither intimidate nor confuse the reader, but rather, will provide a step by step guide of techniques and methods useful for performability model evaluation. In addition, descriptions of certain basic concepts (e.g., reconfiguration and degradation) have been included as an aid to the user.

3.1. System Reliability Examples

As first examples of performability modeling and evaluation, suppose there are only two accomplishment levels a_0 and a_1 where a_0 denotes "success" and a_1 denotes "failure" (in which case performability reduces to reliability).

All of the examples in this section involve systems containing networks of subsystems. Each subsystem can be either "operational" or "failed," and each subsystem fails permanently, that is, once a subsystem has failed, it remains failed for the duration of the utilization period. Subsystems are identical and fail with a constant rate of $\lambda=10^{-4}$/hour. Also, unless otherwise stated, the utilization period length is 10 hours and all subsystems in the system are operational at the start of the utilization period.

3.1.1. TMR System Example

The results with identical inputs (see Figure 3) of three subsystems are voted upon and the majority result is interpreted as the system result--hence, the system is successful if and only if two of the three subsystems work for the entire utilization period. The states of the system will be denoted:

3: Three subsystems operational
2: Two subsystems operational
1: One subsystem operational
0: No subsystems operational.

The transition graph is given in Figure 4. Thus, accomplishment level 0 is attained if the system is in either state 3 or state 2 at the end of the utilization period, while accomplishment level 1 is achieved if the system is in one of the states 1 or 0. Technically,

$$\gamma^{-1}(0) = [\{2,3\}]$$
$$\gamma^{-1}(1) = [\{0,1\}].$$

The P matrix for this system can be generated using type NFAIL transition matrix generation command with 1 group of 3 elements. (See Section 5.4.) Figure 5 shows the METAPHOR session required to compute the performability. One can show that

$$\begin{aligned} p_S(0) &= 3e^{-2\lambda t} - 2e^{-3\lambda t} \\ &= 0.999997 \end{aligned}$$

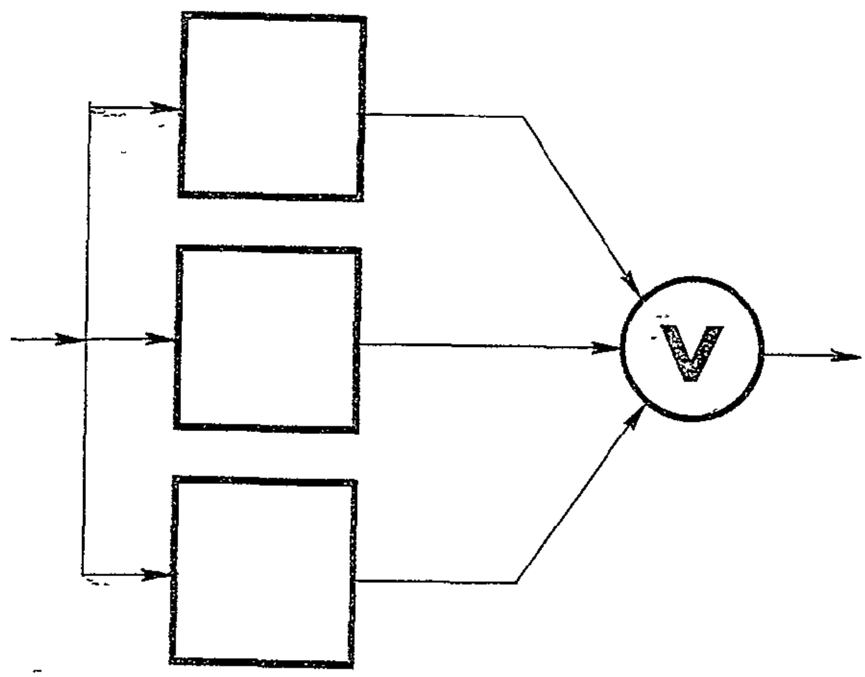


Figure 3. The system of Section 3.1.1., TMR System Example

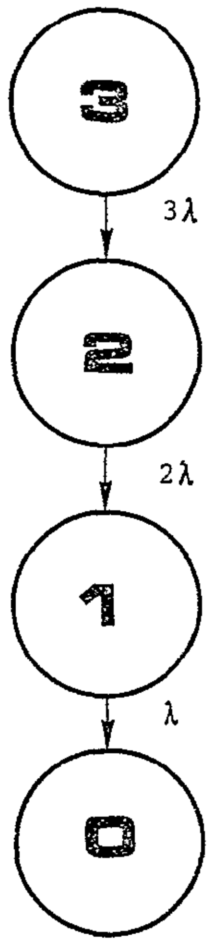


Figure 4. Transition graph for the system of Section 3.1.1., TMR System Example

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE TMR SYSTEM
*** IN SECTION 3.1.1.
***
□:  EVAL

NUMBER OF PHASES?
□:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  3

NUMBER OF CONSTANT BASIC VARIABLES?
□:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
□:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
□:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
□:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
□:  1  1  0  0

ACCOMPLISHMENT LEVEL 1
```

Figure 5. METAPHOR session for the example of Section 3.1.1., TMR System Example

```
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1  1
```

Figure 5. METAPHOR session for the example of Section 3.1.1., TMR System Example-- continued)

$$p_S(1) = 1 - 3e^{-2\lambda t} + 2e^{-3\lambda t}$$
$$= 3\times10^{-6}.$$

3.1.2. TMR System Example with Non-deterministic Initial State

Suppose the system begins its utilization period in some unknown state. In particular, assume the probability of a given subsystem being operational at the start of the utilization period is 0.99. Then the probabilities of beginning the utilization period in a given initial state X_0 are as follows:

$$Pr(X_0=3) = (0.99)^3 = 0.97029$$
$$Pr(X_0=2) = 3(0.99)^2(0.01) = 0.0294$$
$$Pr(X_0=1) = 3(0.99)(0.01)^2 = 0.000297$$
$$Pr(X_0=0) = (0.01)^3 = 10^{-6}$$

Figure 6 illustrates a *METAPHOR* session required to calculate the performability of this system. Analytically,

$$p_S(0) = Pr(X_0=3)(e^{-2\lambda t} - e^{-3\lambda t}) + Pr(X_0=2)e^{-2\lambda t}$$
$$= 0.9996$$
$$p_S(1) = Pr(X_0=3)(1 - 3e^{-2\lambda t} - 2e^{-3\lambda t})$$
$$+ Pr(X_0=2)(1 - e^{-\lambda t}) + Pr(X_0=1) + Pr(X_0=0)$$
$$= 0.00036$$

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** NONDETERMINISTIC INITIAL STATE TMR SYSTEM
*** OF SECTION 3.1.2.
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  3

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  COM
*** LET'S DETERMINE THE INITIAL STATE VECTOR
*** IN THE CALC MODE
***
⎕:  CALC
?
⎕:  UP←.99
0.99
```

Figure 6. METAPHOR session for the example of Section 3.1.2., TMR System Example with Non-deterministic Initial State

```
?
⎕:  UQ←.01
0.01
?
⎕:  U3←UP*3
0.970299
?
⎕:  U2←3×UQ×UP*2
0.029403
?
⎕:  U1←3×UP×UQ*2
0.000297
?
⎕:  U0←UQ*3
1E¯6
?
⎕:  EXIT
EXIT
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  COM
*** HERE WE DEFINE A VARIABLE TO BE THE INITIAL STATE VECTOR
***
⎕:  UI←U3,U2,U1,U0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕: UI
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1  1


PERFORMABILITY FOR THIS MISSION ← 0.9996403467  0.0003596532833
```

Figure 6. METAPHOR session for the example of Section 3.1.2., TMR System Example with Non-deterministic Initial State-- continued)

3.1.3. TMR System with Time-Invariant Basic Variable Example

Consider now the TMR example of the preceding section and suppose the voter works at the beginning of the utilization period with probability 0.99. Suppose too that if the voter works at the beginning then it will work throughout the utilization period, but if the voter fails, the system is unsuccessful. This system can be analyzed with METAPHOR in at least two ways.

1) The initial distribution (I) vector can be modified such that the system begins in state 3 with probability 0.99 and in state 0 with probability 0.01, or

2) A time-invariant basic variable having probability 0.99 can be specified. The trajectory set associated with accomplishment level 0 would have a constant basic variable vector 1 while the vector associated with accomplishment level 1 would be 0.

Let us use the second method for this example. Then the trajectory sets corresponding to each accomplishment level would be:

$$\gamma^{-1}(0) = [\{2,3\}]\times 0$$

$$\gamma^{-1}(1) = [\{0,1\}]\times * \cup [\{2,3\}]\times 1$$

Figure 7 illustrates the session required to perform the calculations. Analytically,

$$p_S(0) = 0.99(3e^{-2\lambda t} - 2e^{-3\lambda t})$$
$$= 0.989997$$

$$p_S(1) = .99(1 - 3e^{-2\lambda t} + 2e^{-3\lambda t}) + .01$$
$$= 0.010003$$

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION  DEMONSTRATES THE EVALUATION OF THE
*** TMR SYSTEM WITH TIME INVARIANT BASIC VARIABLE OF
*** SECTION 3.1.3..
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  3

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  1

PROBABILITIES OF EACH CONSTANT VARIABLE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  0.99

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
```

Figure 7. METAPHOR session for the example of Section 3.1.3., TMR System with Time-Invariant Basic Variable Example

```
⎕:  1  1  0  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY
⎕:  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  2
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1  1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY
⎕:  2
TRAJECTORY SET 2
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY
⎕:  1


PERFORMABILITY FOR THIS MISSION ← 0.9899970349  0.01000296505
```

Figure 7. METAPHOR session for the example of Section 3.1.3., TMR System with Time-Invariant Basic Variable Example-- continued)

3.1.4. Simple Series-Parallel System Example

Next examine the simple series-parallel system shown in Figure 8. The system is successful if and only if a path of operational subsystems from X to Y is present throughout the utilization period. With the intention of employing the NFAIL transition matrix generation facility, partition the system into two groups, each with two subsystems as shown in Figure 9. Then writing the state of the system as the ordered pair (X,Y), where X is the number of operational subsystems in the first group (subsystems A and B) and Y is the number operational in the second group (subsystems C and D), the system is successful if and only if the state at the end of the utilization period is one of

$$(2,2),\ (2,1),\ (2,0),\ (1,2),\ (1,1),\ (0,2),\ \text{or}\ (0,1)\ .$$

Furthermore, the system fails if the state is one of

$$(1,0)\ \text{or}\ (0,0).$$

(See Figure 10.) Thus,

$$\gamma^{-1}(0) = [\{(2,2),(2,1),(2,0),(1,2),(1,1),(0,2),(0,1)\}]$$

$$\gamma^{-1}(1) = [\{(1,0),(0,0)\}]$$

Figure 11 shows the METAPHOR session required to compute the performability. Analytically, one can show that

$$\begin{aligned} p_S(0) &= e^{-4\lambda t} + 4e^{-3\lambda t}(1 - e^{-\lambda t}) + 6e^{-2\lambda t}(1 - e^{-\lambda t}) \\ &\quad + 2e^{-\lambda t}(1 - e^{-\lambda t})^5 \\ &= 0.999999998 \end{aligned}$$

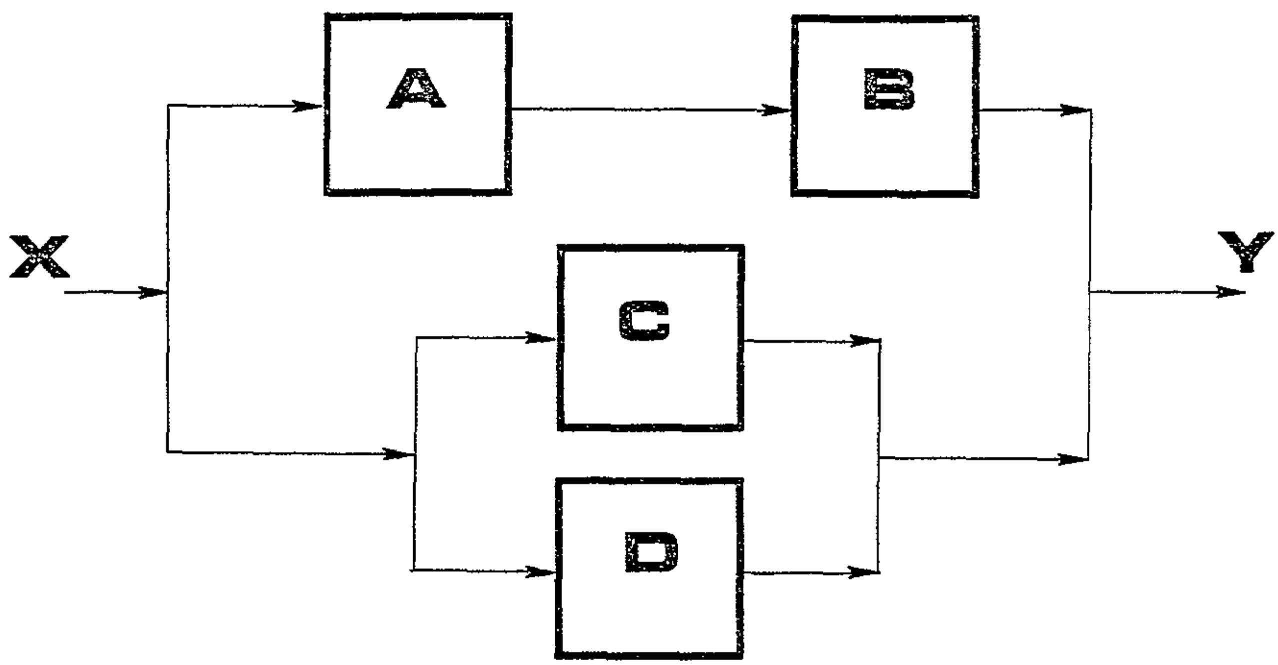


Figure 8. Block diagram of the system in the example of Section 3.1.4., Simple Series-Parallel System Example

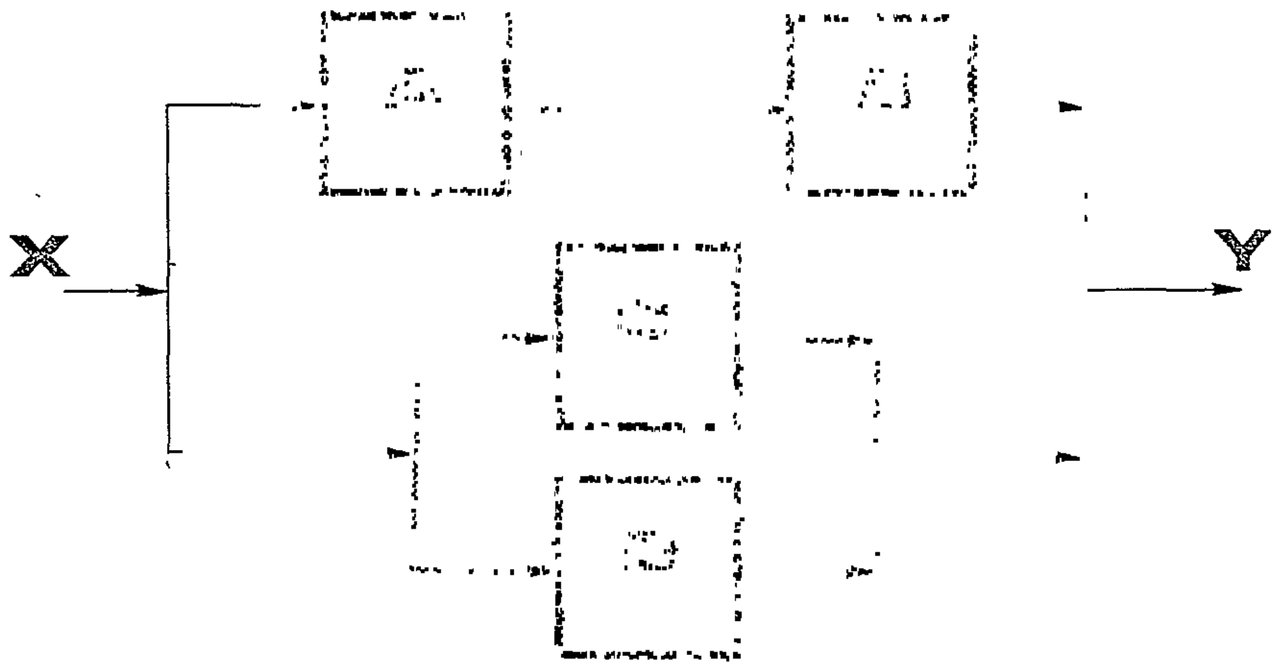


Figure 9. Restructured block diagram of the system in the example of Section 3.1.4., Simple Series-Parallel System Example

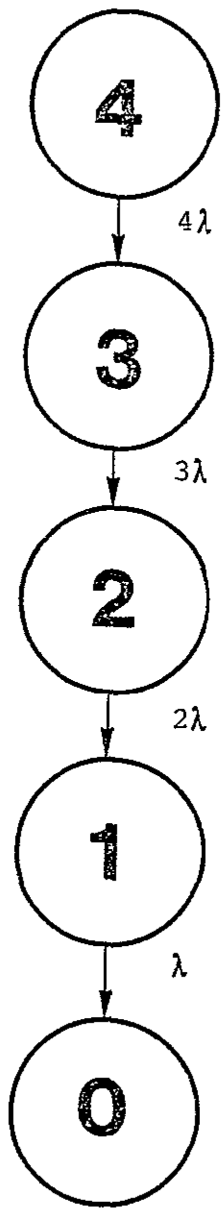


Figure 10. Transition graph for the system of Section 3.1.4., Simple Series-Parallel System Example

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** SERIES¯PARALLEL SYSTEM OF SECTION 3.1.4.
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  9


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  2
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  2  2

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0  0  0  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1  1  1  0  1  1  0

ACCOMPLISHMENT LEVEL 1
```

Figure 11. METAPHOR session for the example of Section 3.1.4., Simple Series-Parallel System Example

NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕: 1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY) :
⎕: 1 0 0 0 0 0 0 0 0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY) :
⎕: 0 0 0 0 0 1 0 0 1

PERFORMABILITY FOR THIS MISSION ← 0.999999998 1.996004496*E*¯9

Figure 11. METAPHOR session for the example of Section 3.1.4., Simple Series-Parallel System Example-- continued)

$$p_s(1) = 1 - (e^{-4\lambda t} + 4e^{-3\lambda t}(1 - e^{-\lambda t}) \\ + 6e^{-2\lambda t}(1 - e^{-\lambda t})^2 + 2e^{-\lambda t}(1 - e^{-\lambda t})^5)$$

$$= 0.000000002$$

3.1.5. Multi-phased Reliability Example

The preceding examples have involved systems wherein the structure of the system has remained the same throughout the utilization period. Consider now a system which is <u>reconfigured</u> at predetermined times, i.e., after fixed periods, the system is restructured such that although individual subsystems retain their identity, their interrelationships are possibly changed. Some commonplace illustrations of reconfiguration are given below:

1) A reconfigurable computing system on board a commercial aircraft. As the aircraft enters various phases (e.g., takeoff, cruise, landing) the computer reconfigures to satisfy the demand for different types of computations based on their cost (in terms of available resources) and criticality.

2) A set of machines in a small factory which produces a set of different products. The system reconfigures when a new manufacturing sequence is introduced to fabricate a new product.

3) An 11 man football team where each athlete plays two positions--one on offense and a second on defense. The system reconfigures when the team either gains or loses possession of the ball.

4) A corporation's management. Here the system reconfigures during a "shakeup" of the corporation's organization.

5) An automobile with a spare tire. When one tire goes flat, the system reconfigures to an automobile with no spare tire.

Systems which involve reconfiguration after deterministic periods of time are said to be phased (see [16]). For example, the system in Figure 12 involves three subsystems in series during the first 10 time of units of the utilization period, i.e., the first phase. Then however, those same subsystems are restructured into a parallel configuration for the final 10 time units (the second phase). Again, the system is successful if and only if a path of operational subsystems from X to Y is present throughout the utilization period. We shall write the state of the system as the number of operational subsystems, and we shall sample state trajectories at the end of both phases. Now, suppose we have determined that a system is successful if

i) the state at the end of phase 1 is 3, and
ii) the state at the end of phase 2 is at least 1.

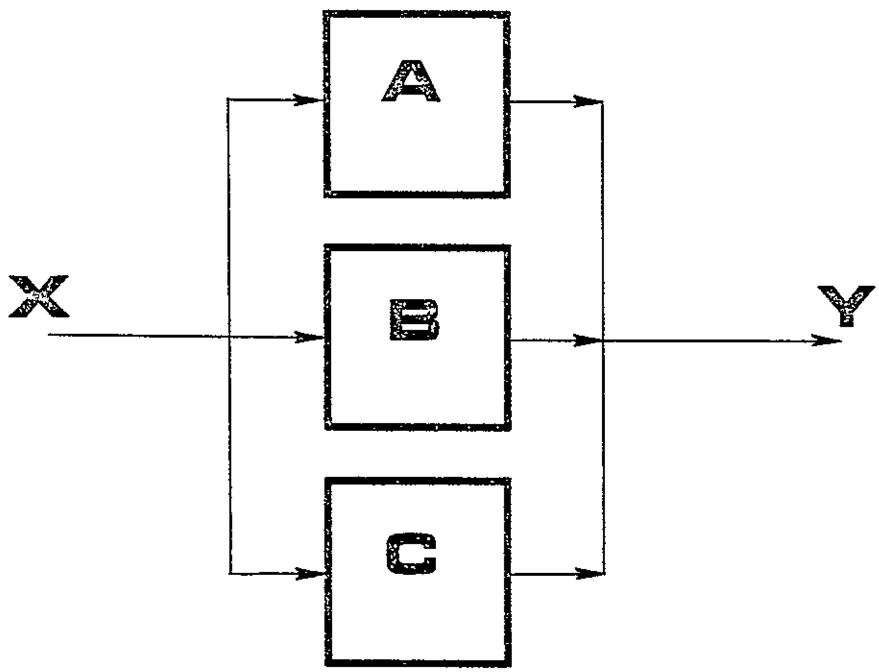
Thus,

$$\gamma^{-1}(0) = [3\ \{1,2,3\}]$$

and,

$$\gamma^{-1}(1) = [\{0,1,2\}\ *] \cup [3\ 0].$$

The interphase transition matrix between phases 1 and 2 is the identity matrix because the state of the system (i.e., the number of operational subsystems) remains unchanged when the phase changes. For the first intraphase transition matrix, the NFAIL matrix generation command can be used since we are interested in the actual number (1, 2, or 3) of operational subsystems. However, because the second phase requires only knowledge of whether all 3 subsystems work, the DEDFAIL command (with 1 group of 3 subsystems) may be employed. Figure 13 illustrates a session with METAPHOR to calculate the system



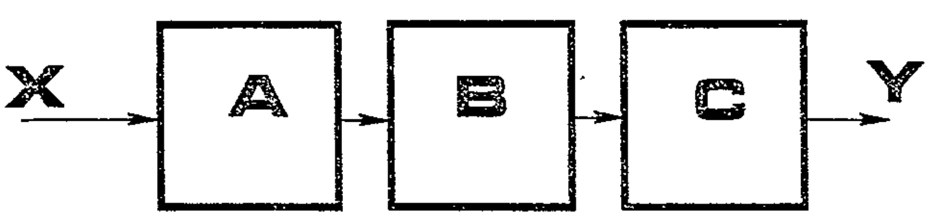


Figure 12. Block diagram of the system in the example of Section 3.1.5., Multi-phased Reliability Example

3.1.5. Multi-phased Reliability Example

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** MULTIPHASED SYSTEM IN SECTION 3.1.5.
***
□:  EVAL

NUMBER OF PHASES?
□:  2

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  4  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  3

PHASE 2:

WHAT TYPE OF P MATRIX?
□:  DEDFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1¯2:

WHAT TYPE OF H MATRIX?
□:  IDENTITY
```

Figure 13. METAPHOR session for the example of Section 3.1.5., Multi-phased Reliability Example

```
NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  2
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  1  1  1
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1  1
TRAJECTORY SET 2
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  0  1


PERFORMABILITY FOR THIS MISSION ← 0.9970034995  0.002996500505
```

Figure 13. METAPHOR session for the example of Section 3.1.5., Multi-phased Reliability Example-- continued)

performability. We can show analytically that

$$p_S(0) = e^{-6\lambda t} + 3e^{-5\lambda t}(1 - e^{-\lambda t}) + 3e^{-4\lambda t}(1 - e^{-\lambda t})^2$$
$$= 0.997$$
$$p_S(1) = 1 - e^{-6\lambda t} - 3e^{-5\lambda t}(1 - e^{-\lambda t}) - 3e^{-4\lambda t}(1 - e^{-\lambda t})^2$$
$$= 0.003 .$$

where t = 10 hours.

3.1.6. "Advanced" Series-Parallel System Example

Consider the system in Figure 14. The system is successful if and only if a path of operational subsystems from X to Y is present throughout the entire utilization period.

This system is "advanced" only in the sense that presently no built-in function within *METAPHOR* can automatically generate the transition matrix P. Hence the user may do one of the following: 1) construct an *APL* function for calculating the matrix, or 2) calculate the matrix and enter it via the *GIVEN* command (see Section 5.2.), or finally, 3) find an equivalent system having the same performability, such that the transition matrices associated with the system can be generated by METAPHOR. So, making analogies with the technique of Esary and Ziehms [16] for analyzing phased missions, the user can "phase" the model as shown in Figure 15. The analogous system has two subsystems operating over two 10 hour phases. Thus, the *NFAIL* transition matrix generation (for a single group of 2 subsystems) can be employed for both phases. Also, the *IDENTITY* matrix generator should be used for the interphase transition matrix. However, caution must be used when utilizing this third

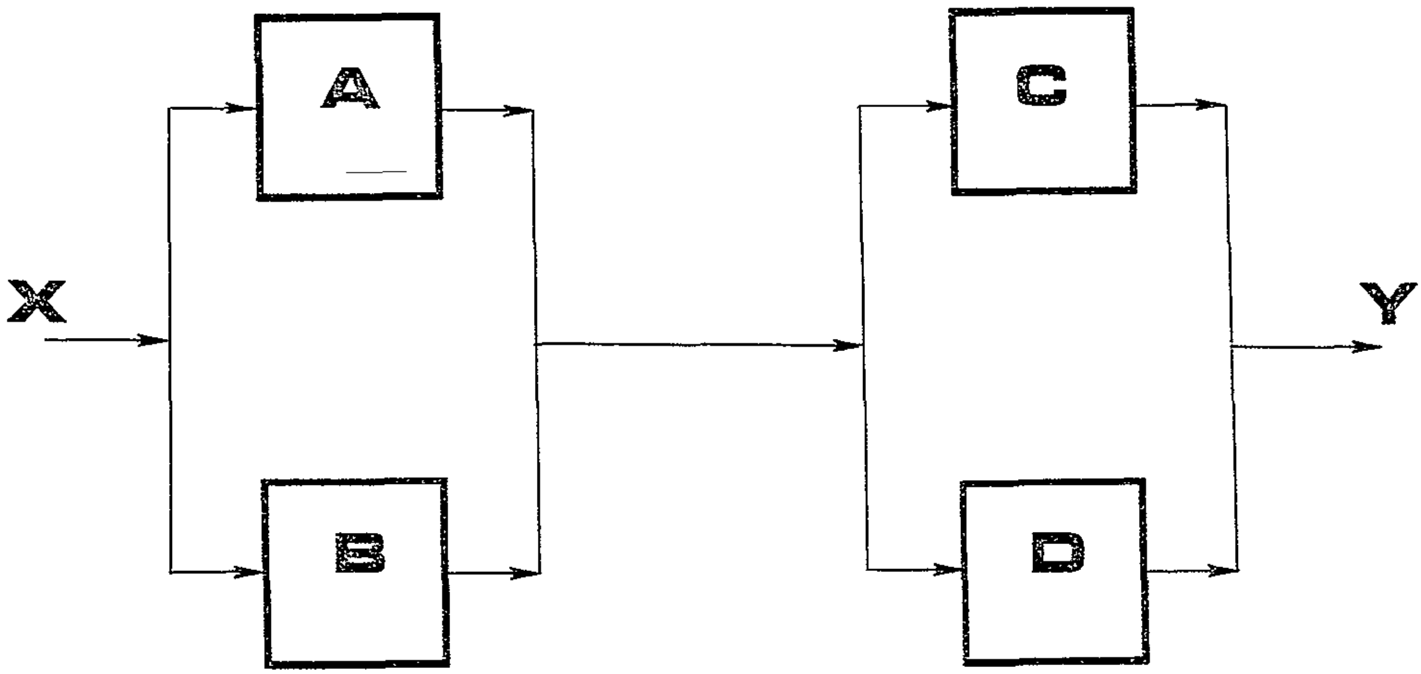


Figure 14. Block diagram of the system in the example of Section 3.1.6., "Advanced" Series-Parallel System Example

3.1.6. "Advanced" Series-Parallel System Example

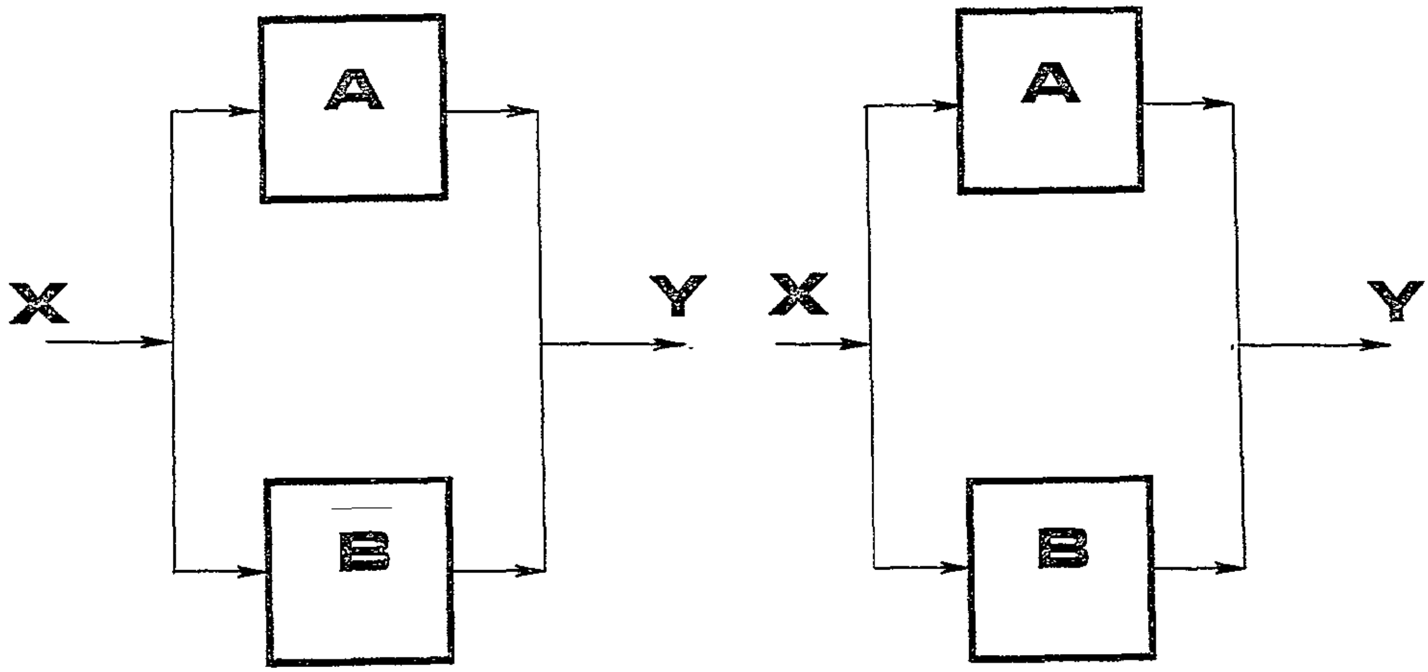


Figure 15. System in the example of Section 3.1.6. ("Advanced" Series-Parallel System Example) restructured into two phases

approach, for the underlying theory must be well understood. For example, usually only stationary (or wide-sense stationary) distributions can be so analyzed unless proper compensations are made.

If the state of the augmented system is written as the number of subsystems operating, then the system is successful if and only if the state is at least 1 throughout the utilization period. We shall write state trajectories as [a, b] where a and b are the states at the ends of phases 1 and 2 respectively. Accomplishment level 0 then corresponds to having state trajectories of

[2 2], [2 1], [1 2], or [1 1]

while accomplishment level 1 occurs for trajectories

[2 0], [1 0], [0 0], [0 1], or [0 2].

That is

$$\gamma^{-1}(0) = [* \ \{1,2\}]$$

$$\gamma^{-1}(1) = [* \ 0] \cup [0 \ \{1,2\}].$$

A METAPHOR session for analyzing the system's performability is shown in Figure 16. Analytically, we can show

$$p_S(0) = e^{-4\lambda t} + 4e^{-3\lambda t}(1 - e^{-\lambda t}) + 4e^{-2\lambda t}(1 - e^{-\lambda t})^2$$

$$= 0.999998$$

$$p_S(1) = 1 - e^{-4 \ t} - 4e^{-3\lambda t}(1 - e^{-\lambda t}) - 4e^{-2\lambda t}(1 - e^{-\lambda t})^2$$

$$= 0.000002 \ .$$

3.1.6. "Advanced" Series-Parallel System Example

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** 'ADVANCED' SERIES-PARALLEL SYSTEM OF
*** SECTION 3.1.6.
***
□:  EVAL

NUMBER OF PHASES?
□:  2

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  3  3


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  2

PHASE 2:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  2


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME
```

Figure 16. METAPHOR session for the example of Section 3.1.6., "Advanced" Series-Parallel System Example

```
PHASE 1¯2:

WHAT TYPE OF H MATRIX?
⎕:  IDENTITY

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  2
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1
TRAJECTORY SET 2
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0


PERFORMABILITY FOR THIS MISSION ← 0.999996008  3.992009325E¯6
```

Figure 16. METAPHOR session for the example of Section 3.1.6., "Advanced" Series-Parallel System Example-- continued)

3.2. Systems with Degradable Performance

The next set of systems to be considered are those whose performance is degradable. These are systems which can display many different levels of performance (from the user's viewpoint) during the utilization period, depending on the history of the system's structure and environment. In particular, we refer here to systems for which the performance can not be simply classified as either "success" or "failure." Some common systems that illustrate degradability are listed below:

1) A reconfigurable computing system on board a commercial aircraft. As subsystems (modules fail, the ability of the system to perform various computations will be decreased.

2) A set of machines in a small factory. As machines breakdown, the ability of the factory to produce a product will be decreased; key machines may bring the entire factory to a stop, while other machines reduce throughput, increase costs, etc.

3) A football team. The system degrades as athletes grow tired or are injured.

4) An automobile with pneumatic tires. As air leaks from the tires, the driving efficiency (in terms of miles per gallon of gasoline) of the car decreases.

3.2.1. Simple Degradable System Over a Single Phase

Consider the simple degradable system shown in Figure 17. The system is constructed of two subsystems: A having a high throughput and B having a low one. If subsystem A works during the entire utilization period (phase), the system performance is considered excellent; denote this situation accomplishment level 0. However, should A fail while B works for the entire

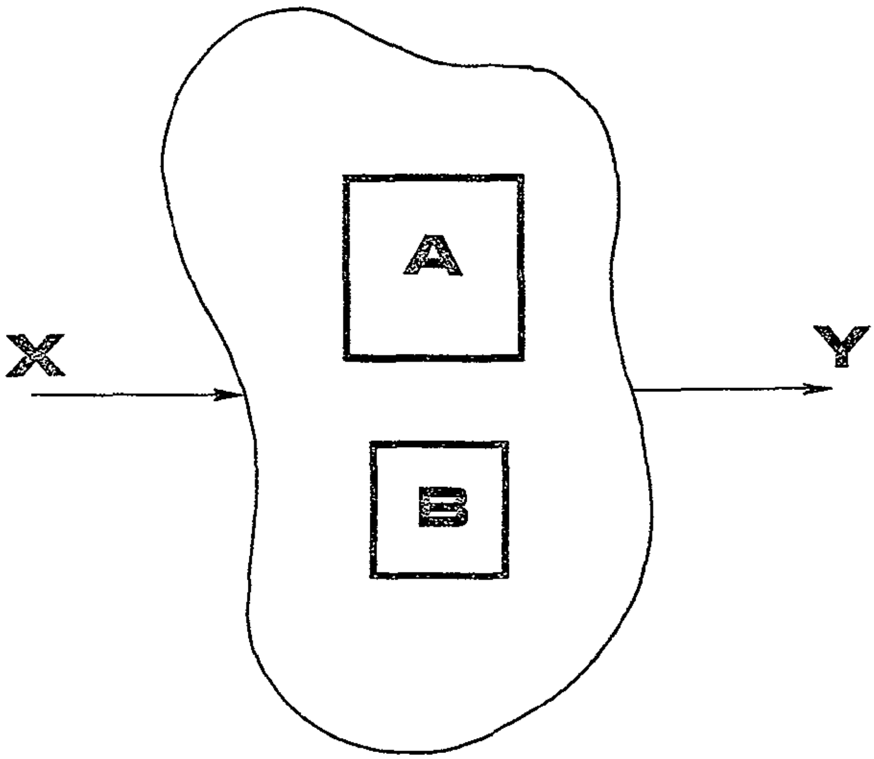


Figure 17. Block diagram of the system in the example of Section 3.2.1., Simple Degradable System Over a Single Phase

3.2.1. Simple Degradable System Over a Single Phase utilization period, the system performance is deemed acceptable, but poor; call this case accomplishment level 1. Finally, if both A and B fail during the utilization period, the performance is unacceptable; call this situation accomplishment level 2.

We shall denote the state of the system as the ordered pair (a,b) where a and b are binary and

$$a = \begin{cases} 0 & \text{if A is operational} \\ 1 & \text{if A is failed} \end{cases}$$

$$b = \begin{cases} 0 & \text{if B is operational} \\ 1 & \text{if B is failed.} \end{cases}$$

Hence, sampling state trajectories at the end of the utilization period, accomplishment level 0 is achieved if the state trajectory is one of

$$[(1,1)] \text{ or } [(1,0)],$$

accomplishment level 1 if the trajectory is

$$[(0,1)],$$

and level 2 if the trajectory is

$$[(0,0)].$$

Thus,

$$\gamma^{-1}(0) = [\{(1,1),(1,0)\}]$$
$$\gamma^{-1}(1) = [(0,1)]$$
$$\gamma^{-1}(2) = [(0,0)].$$

The intraphase transition matrix P can be computed using the NFAIL command since the state of both subsystems must be known. Figure 18 demonstrates a METAPHOR session for computing the systems's performability. One can show analytically that

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** SINGLE PHASED SIMPLE DEGRADABLE SYSTEM OF SECTION 3.2.1.
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  2
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  1  1

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  3

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0  0

ACCOMPLISHMENT LEVEL 1
```

Figure 18. METAPHOR session for the example of Section 3.2.1., Simple Degradable System Over a Single Phase

```
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1   0   0   0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0   0   1   0

ACCOMPLISHMENT LEVEL 2
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1   0   0   0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0   0   0   1


PERFORMABILITY FOR THIS MISSION ← 0.9990004998  0.000998501166  9.990005831E¯7
```

Figure 18. METAPHOR session for the example of Section 3.2.1., Simple Degradable System Over a Single Phase-- continued)

$$p_S(0) = e^{-\lambda t} = 0.999$$
$$p_S(1) = e^{-\lambda t}(1 - e^{-\lambda t}) = 0.000998$$
$$p_S(2) = (1 - e^{-\lambda t})^2 = 0.000000999 .$$

3.2.2. Simple Degradable System over Multiple Phases

Next we study the simple degradable system shown in Figure 19. As in Section 3.2.1., the system has two subsystems, a high throughput subsystem A and a low throughput subsystem B. However, here the utilization period is divided into two phases, each of length 10 hours. Qualitatively, phase 1 is deemed very important, and the throughput of both subsystems A and B is required to perform the phase well; hence, the choice of accomplishment levels emphasizes phase 1. Below are the 4 levels along with their definitions:

level 0: A and B work throughout the utilization period
level 1: A and B work throughout phase 1; A or B work throughout phase 2
level 2: A and B work throughout phase 1; neither A nor B works throughout phase 2
level 3: Neither A nor B work throughout phase 1.

Writing states and trajectories as in Section 3.2.1., we have

$$\gamma^{-1}(0) = [(1,1)\ (1,1)]$$
$$\gamma^{-1}(1) = [(1,1)\ \{(0,1),(1,0)\}]$$
$$\gamma^{-1}(2) = [(1,1)\ (0,0)]$$
$$\gamma^{-1}(3) = [\{(1,0),(0,1),(0,0)\}\ *] .$$

Because the system does not reconfigure between phases, the state of the system remains the same when the phase changes, so the interphase transition (H) matrix is the identity matrix; the

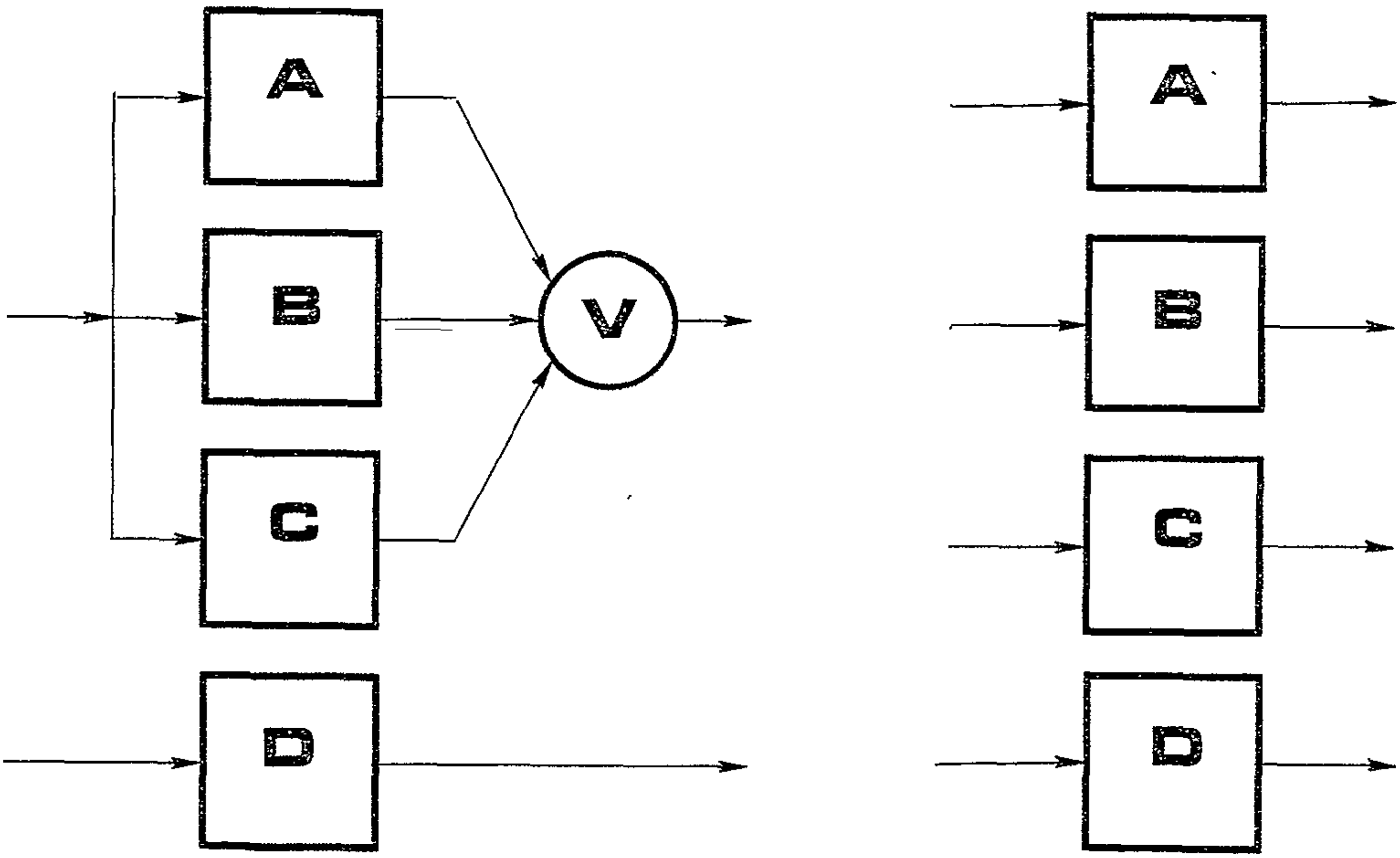


Figure 19. Block diagram of the system in the example of Section 3.2.2.

IDENTITY command can be employed here. The intraphase transition (P) matrices are both the same as the one in Section 3.2.1., and the NFAIL command is used to generate them. Figure 20 illustrates a METAPHOR session to compute the performability of the system. Letting t=10 hours, then analytically,

$$p_S(0) = e^{-4\lambda t} = 0.996$$

$$p_S(1) = 2e^{-3\lambda t}(1 - e^{-\lambda t}) = 0.001993$$

$$p_S(2) = e^{-2\lambda t}(1 - e^{-\lambda t})^2 = 0.00000097$$

$$p_S(3) = 2e^{-\lambda t}(1 - e^{-\lambda t}) + (1 - e^{-\lambda t})^2 = 0.001998 .$$

3.2.4. Degradable System Over Multiple Phases

Consider next the system in Figure 21. Here we have 4 subsystems being employed in 2 phases, each 10 hours in length. In phase 1, subsystems A, B, and C are configured in a TMR manner while subsystem D is standing alone, and in phase 2, all 4 subsystems are standing alone. There are four accomplishment levels:

- level 0: In phase 1, either D or the TMR configuration (ABC) must work (i.e., 2 of the 3 subsystems A, B, or C) must work, and in phase 2, subsystems A, B, and D must work.
- level 1: In phase 1, either D or the TMR configuration (ABC) must work, and in phase 2, subsystems A, C, and D must work, but B must not work.
- level 2: In phase 1, either D or the TMR configuration (ABC) must work, and in phase 2, subsystems A and D must work, but B and C must not work.
- level 3: In phase 1, both D and 2 of the other three subsystems must fail, or in phase 2, A or D must fail.

The states of the first phase could be written in one of two

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** MULTI-PHASED SIMPLE DEGRADABLE SYSTEM OF
*** SECTION 3.2.2.
***
□:  EVAL

NUMBER OF PHASES?
□:  2

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  4  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  2
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  1  1

PHASE 2:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  2
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  1  1


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME
```

Figure 20. METAPHOR session for the example of Section 3.2.2., Simple Degradable System over Multiple Phases

```
PHASE 1¯2:

WHAT TYPE OF H MATRIX?
⎕:  IDENTITY

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  4

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  1  1  0

ACCOMPLISHMENT LEVEL 2
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  0  1

ACCOMPLISHMENT LEVEL 3
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
```

Figure 20. METAPHOR session for the example of Section 3.2.2., Simple Degradable System over Multiple Phases-- continued)

```
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  0  0  0
PHASE 1:
ENTER THE G DIAGONAL ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  1  1  1
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  1  1



PERFORMABILITY FOR THIS MISSION ← 0.9960079893  0.001993012319  9.970045786E¯7
      0.001998001333
```

Figure 20. METAPHOR session for the example of Section 3.2.2., Simple Degradable System over Multiple Phases-- continued)

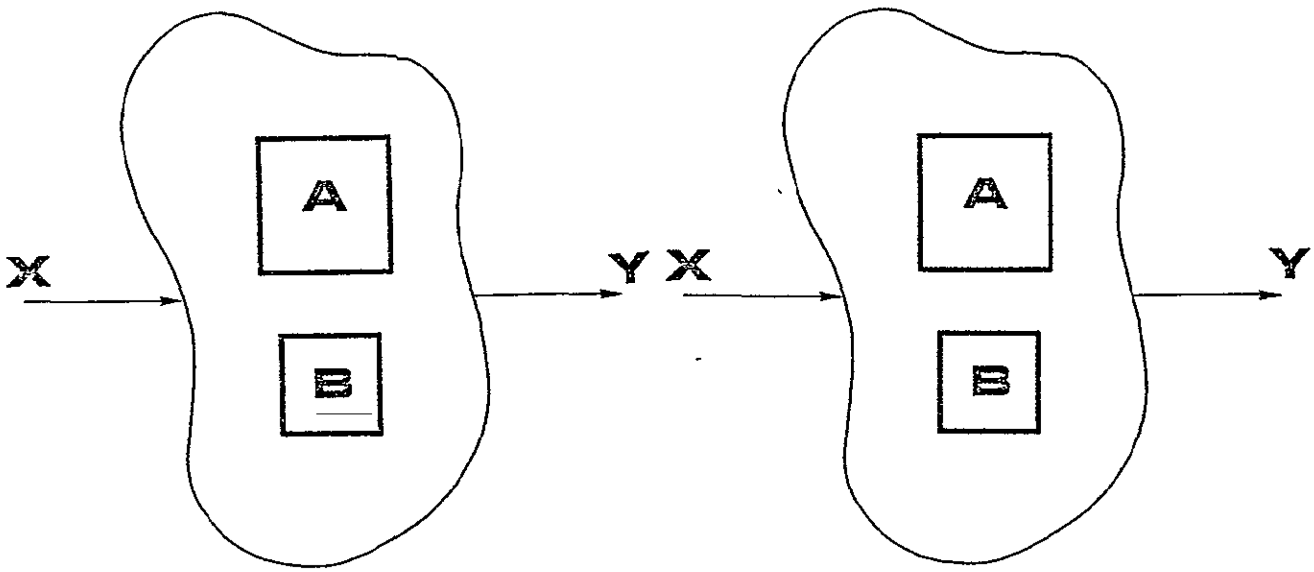


Figure 21. Block diagram of the system in the example of Section 3.2.4., Degradable System Over Multiple Phases

3.2.4. Degradable System Over Multiple Phases ways:

1) The status of each subsystem can be tracked separately, e.g., (C_A, C_B, C_C, C_D) where for $i \varepsilon \{A, B, C, D\}$

$$C_i = \begin{cases} 1 & \text{if subsystem } i \text{ is operational throughout the phase} \\ 0 & \text{otherwise} \end{cases}$$

This representation requires 16 states.

2) The status of subsystem D and the number of subsystems in the TMR configuration (ABC) are tracked, e.g., (C_{ABC}, C_D), where,

$$C_{ABC} = k, \quad 0 \leq k \leq 3 \text{ where } k \text{ is the number of subsystems in } \{A, B, C\} \text{ which are operational at the end of the phase}$$

$$C_D = \begin{cases} 1 & \text{if subsystem D is operational throughout the phase} \\ 0 & \text{otherwise} \end{cases}$$

This representation requires only 8 states.

Although the second method requires somewhat more computation, the analysis of trajectory sets is simpler since the number of states has been reduced; moreover, inputting these trajectory sets to METAPHOR is easier.

We shall employ the second method. Then from the above descriptions of the accomplishment levels, we have

$$\gamma^{-1}(0) = [\{(3,1),(3,0),(2,1),(2,0),(1,1),(0,1)\} \\ (1,1,1,1),(1,1,0,1)\}]$$

$$\gamma^{-1}(1) = [\{(3,1),(3,0),(2,1),(2,0),(1,1),(0,1)\} \\ (1,1,1,1),(1,0,1,1)\}]$$

$$\gamma^{-1}(2) = [\{(3,1),(3,0),(2,1),(2,0),(1,1),(0,1)\} \\ (1,1,1,1),(1,0,0,1)\}]$$

$$\gamma^{-1}(3) = [\{(1,0),(0,0)\} *] \cup [\{(3,1),(3,0),(2,1),(2,0),(1,1),(0,1)\} \{(1,1,1,0,(1,1,0,0),(1,0,1,0),(1,0,0,0,),(0,1,1,1),(0,1,1,0),(0,1,0,1),(0,0,1,1),(0,0,1,0),(0,0,0,1),(0,0,0,0)\}]$$

Phase 2 states could be denoted by method 1) above. If phase 1 states and phase 2 states are denoted by method 1), then the interphase transition (H) matrix would be the identity. However, if phase 1 states are represented by method 2), then the interphase transition (H) matrix must be computed based on the initial state distribution (I vector), subsystem failure rates, (λ), and phase 1 duration. Figure 22 shows the H matrix for this example.

Now, for phase 1, the interphase transition (P) matrix must keep track of only the number of operational subsystems--hence the NFAIL command (see Section 5.4.) with 2 groups (the first having 3 subsystems and the second having 1 subsystem) can be used to generate the interphase transition (P) matrix. However, for phase 2, the P matrix must keep tabs on which specific subsystems are operating; hence the *DEDFAIL* command (see Section 5.1.) is appropriate. A *METAPHOR* session illustrating the analysis of this example is presented in Figure 23.

Analytically,

$$p_S(0) = e^{-6\lambda t} = 0.994$$

$$p_S(1) = e^{-7\lambda t}(1 - e^{-\lambda t}) + e^{-6\lambda t}(1 - e^{-\lambda t}) = 0.001986$$

$$p_S(2) = e^{-6\lambda t}(1 - e^{-\lambda t})^2 + 2e^{-5\lambda t}(1 - e^{-\lambda t})^2$$

3.2.4. Degradable System Over Multiple Phases

$$
H(0) = \begin{bmatrix}
i(q_{(31)}) \\
i(q_{(31)}) + i(q_{(30)}) \\
i(q_{(31)}) + i(q_{(21)}) \\
i(q_{(31)}) + i(q_{(30)}) + i(q_{(21)}) + i(q_{(20)}) \\
i(q_{(31)}) + i(q_{(21)}) + i(q_{(11)}) \\
i(q_{(31)}) + i(q_{(30)}) + i(q_{(21)}) + i(q_{(20)}) + i(q_{(11)}) + i(q_{(10)}) \\
i(q_{(31)}) + i(q_{(21)}) + i(q_{(11)}) + i(q_{(01)}) \\
1
\end{bmatrix}^T
$$

$$
\begin{array}{c}
 \\
(31) \\ (30) \\ (21) \\ (20) \\ (11) \\ (10) \\ (01) \\ (00)
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccccccc}
(1111) & (1110) & (1101) & (1100) & (1011) & (1010) & (1001) & (1000) & (0111) & (0110) & (0101) & (0100) & (0011) & (0010) & (0001) & (0000)
\end{array} \\
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\end{array}
$$

Figure 22. Interphase transition (H) matrix for the example of Section 3.2.4., Degradable System Over Multiple Phases

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** DEGRADABLE SYSTEM OVER MULTIPLE PHASES EXAMPLE
*** OF SECTION 3.2.4.
***
□:  EVAL

NUMBER OF PHASES?
□:  2

NUMBER OF STATES PER PHASE?  (SPACE BETWEEN EACH NUMBER)
□:  8  16


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  2
ENTER NUMBER OF COMPONENTS PER GROUP  (SPACE BETWEEN EACH NUMBER):
□:  3  1

PHASE 2:

WHAT TYPE OF P MATRIX?
□:  DEDFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1¯2:

WHAT TYPE OF H MATRIX?
□:  GIVEN
```

Figure 23. METAPHOR session for the example of Section 3.2.4., Degradable System Over Multiple Phases

```
ENTER THE MATRIX, 1 ROW AT A TIME

ROW 1:
⎕:    1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
ROW 2:
⎕:    0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
ROW 3:
⎕:    0  0  0.3333333333  0  0.3333333333  0  0  0  0.3333333333
      0  0  0  0  0  0  0
ROW 4:
⎕:    0  0  0  0.3333333333  0  0.3333333333  0  0  0
      0.3333333333  0  0  0  0  0  0
ROW 5:
⎕:    0  0  0  0  0  0  0.33333333333  0  0  0  0.33333333333
      0  0.33333333333  0  0  0
ROW 6:
⎕:    0  0  0  0  0  0  0  0.3333333333  0  0  0  0.3333333333  0
      0.3333333333  0  0
ROW 7:
⎕:    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0
ROW 8:
⎕:    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕:  4

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  1  0  0  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  1  0  0  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
```

Figure 23. METAPHOR session for the example of Section 3.2.4., Degradable System Over Multiple Phases-- continued)

```
⎕:  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0

ACCOMPLISHMENT LEVEL 2
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  1  0  0  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0

ACCOMPLISHMENT LEVEL 3
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  4
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  1  0  1  0  1  0  1
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
TRAJECTORY SET 2
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  0  1  0  1  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
TRAJECTORY SET 3
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  1  0  0  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  1  0  1  0  1  0  1  0  1  0  1  0  1  0  1
TRAJECTORY SET 4
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  1  0  0  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  0  0  0  0  0  1  0  1  0  1  0  1  0
```

Figure 23. METAPHOR session for the example of Section 3.2.4., Degradable System Over Multiple Phases-- continued)

```
PERFORMABILITY FOR THIS MISSION ← 0.9940179641  0.001986049217  2.981060619E¯6
      0.003993005669
```

Figure 23. METAPHOR session for the example of Section 3.2.4., Degradable System Over Multiple Phases-- continued)

$$= 0.00000298$$

$$p_S(3) = (1-e^{-\lambda t}) + 2e^{-4\lambda t}(1 - e^{-\lambda t}) + e^{-5\lambda t}(1 - e^{-\lambda t})$$
$$+ 3e^{-2\lambda t}(1 - e^{-\lambda t})^2 + e3^{-3\lambda t}(1 - e^{-\lambda t})^2$$
$$+ 2e^{-4\ t}(1 - e^{-\lambda t})^2 + 2e^{-\lambda t}(1 - e^{-\lambda t})^3$$
$$= 0.003993$$

3.2.3. The Degrading Processor Model of the Third Semi-Annual Status Report

In the third Semi-Annual Status Report [4], an air transport mission was modeled; three base models were considered: a dedicated processor model, a dedicated group processor model, and a degrading model. We consider here the latter model, whose transition graph is shown in Figure 24. The system modeled here consists of four subsystems having failure rates of 10^{-5}/hour and the state of the system denotes the number of operational subsystems. There are three phases of lengths 2.5 hours, 2.5 hours, and 0.5 hours respectively. In addition, there is a time-invariant base variable (weather). Five accomplishment levels are distinguished, each corresponding to the trajectory sets below. (See [4] for their derivation.)

$$\begin{aligned}\gamma^{-1}(0) = {} & ([\{4,3,2\}\ \{4,3\}\ \{4,3\}] \quad [*]) \times \\ & \cup\ ([\{4,3,2\}\ \{4,3\}\ 1] \times [0]) \\ & \cup\ ([\{4,3,2\}\ 2\ \{4,3,1\}] \times [0])\end{aligned}$$

$$\begin{aligned}\gamma^{-1}(1) = {} & ([\{1,0\}\ \{4,3,2\}\ \{4,3,1\}] \quad [0]) \times \\ & \cup\ ([\{4,3,2\}\ \{1,0\}\ \{4,3,1\}] \times [0]) \\ & \cup\ ([\{4,3,2\}\ \{4,3\}\ 2] \times [*]) \\ & \cup\ ([\{4,3,2\}\ \{4,3\}\ 0] \times [0]) \\ & \cup\ ([\{4,3,2\}\ 2\ \{2,0\}] \times [0])\end{aligned}$$

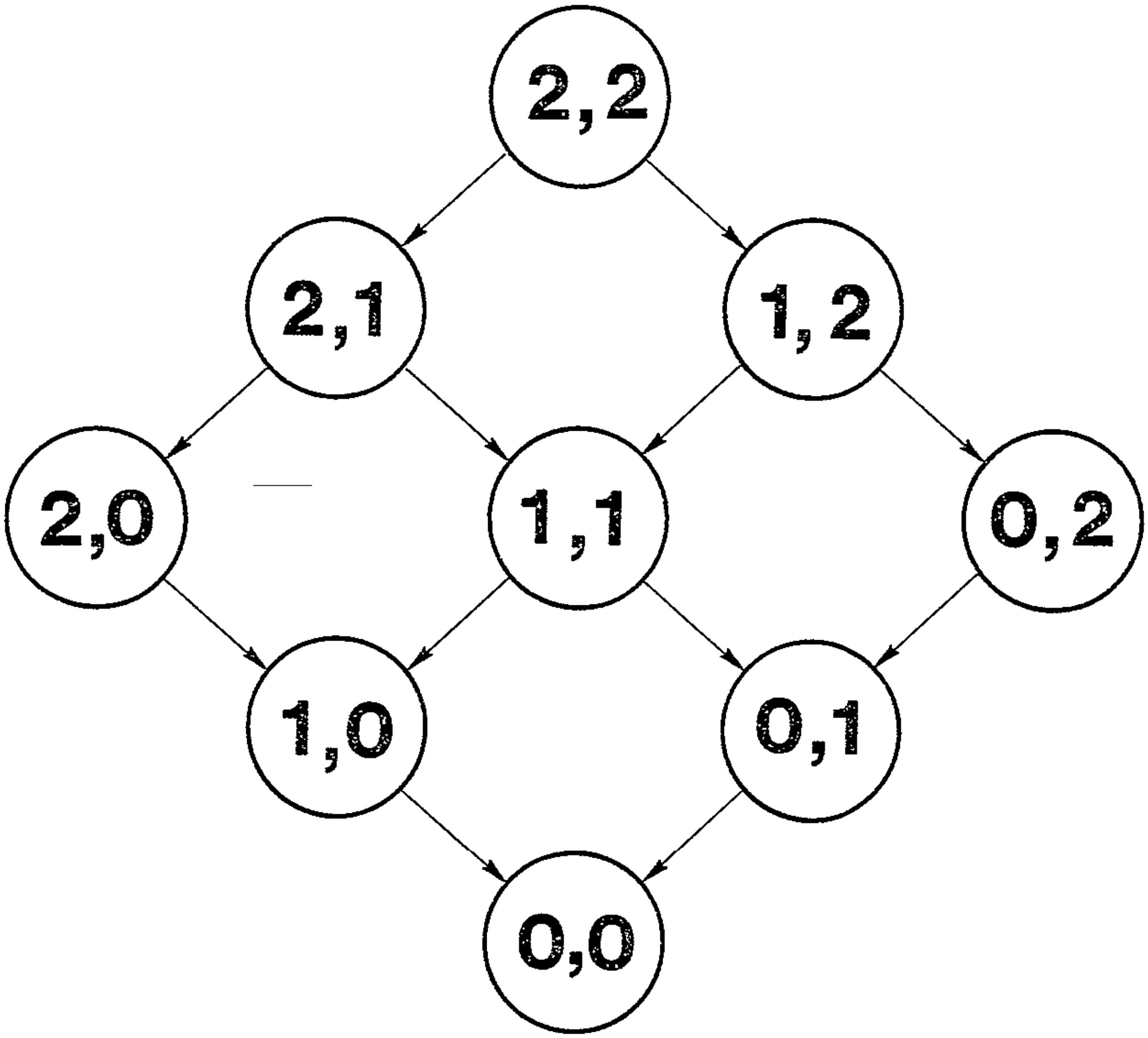


Figure 24. The transition graph for the example of Section 3.2.3., The Degrading Processor Model of the Third Semi-Annual Status Report

$$\gamma^{-1}(2) = ([\{4,3,2\}\ 2\ \{4,3,1\}] \quad [1]) \times$$

$$\begin{aligned}\gamma^{-1}(3) = {} & ([\{1,0\}\ \{4,3,2\}\ \{4,3,1\}] \quad [*]) \times \\ & \cup ([\{4,3,2\}\ \{1,0\}\ \{4,3,1\}] \times [1]) \\ & \cup ([\{4,3,2\}\ 2\ \{2,0\}] \times [1])\end{aligned}$$

$$\begin{aligned}\gamma^{-1}(4) = {} & ([\{1,0\}\ \{1,0\}\ *] \quad [*]) \times \\ & \cup ([\{1,0\}\ \{4,3,2\}\ \{2,0\}] \times [*]) \\ & \cup ([\{4,3,2\}\ \{1,0\}\ \{2,0\}] \times [*]) \\ & \cup ([\{4,3,2\}\ \{4,3\}\ \{1,0\}] \times [1])\end{aligned}$$

The intraphase transition (P) matrix between each phase is the identity matrix (generated with the IDENTITY command), while the interphase transition matrices can be generated with the NFAIL command. A session determining the performability of the system is shown in Figure 25.

3.2.3. The Degrading Processor Model of the Third Semi-Ann Status Report

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE EVALUATION OF THE
*** DEGRADING SYSTEM OF THE THIRD SEMI¯ANNUAL
*** STATUS REPORT. THE EXAMPLE IS DISCUSSED IN SECTION 3.2.3.
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  3

NUMBER OF STATES PER PHASE? (SPACE BETWEEN EACH NUMBER)
⎕:  5  5  5


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  2.5
ENTER COMPONENT FAILURE RATE
⎕:  1E¯5
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  (SPACE BETWEEN EACH NUMBER):
⎕:  4

PHASE 2:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  2.5
ENTER COMPONENT FAILURE RATE
⎕:  1E¯5
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  (SPACE BETWEEN EACH NUMBER):
⎕:  4

PHASE 3:

WHAT TYPE OF P MATRIX?
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.

```
⎕: NFAIL
ENTER PHASE LENGTH
⎕: 0.5
ENTER COMPONENT FAILURE RATE
⎕: 0.0001
ENTER NUMBER OF GROUPS
⎕: 1
ENTER NUMBER OF COMPONENTS PER GROUP  (SPACE BETWEEN EACH NUMBER):
⎕: 4


SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1¯2:

WHAT TYPE OF H MATRIX?
⎕: IDENTITY

PHASE 2¯3:

WHAT TYPE OF H MATRIX?
⎕: IDENTITY

NUMBER OF CONSTANT BASIC VARIABLES?
⎕: 1

PROBABILITIES OF EACH CONSTANT VARIABLE?  (SPACE BETWEEN EACH NUMBER )
⎕: 0.019

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕: 5

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕: 3
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1  1  0  0  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕: 2
TRAJECTORY SET 2
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

3.2.3. The Degrading Processor Model of the Third Semi-Annual Status Report

```
⎕: 1 0 0 0 0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 1 0 0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 0 0 0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 0 0 0 1 0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR (SPACE BETWEEN EACH ENTRY)
⎕: 0
TRAJECTORY SET 3
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1 0 0 0 0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 1 0 0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 0 0 1 0 0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 0 1 0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR (SPACE BETWEEN EACH ENTRY)
⎕: 0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕: 5
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1 0 0 0 0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 0 0 0 1 1
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 1 0 0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 0 1 0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR (SPACE BETWEEN EACH ENTRY)
⎕: 0
TRAJECTORY SET 2
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕: 1 0 0 0 0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕: 1 1 1 0 0
PHASE 2:
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

```
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  1  1
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  1  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY):
⎕:  0
TRAJECTORY SET 3
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  1  0  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY):
⎕:  2
TRAJECTORY SET 4
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  0  1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY):
⎕:  0
TRAJECTORY SET 5
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  1  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  1  0  1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY):
⎕:  0

ACCOMPLISHMENT LEVEL 2
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

```
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  1  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  1  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕:  1

ACCOMPLISHMENT LEVEL 3
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  3
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  1  1
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  1  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕:  1
TRAJECTORY SET 2
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  1  1
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  1  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕:  1
TRAJECTORY SET 3
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

```
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  1   1   1   0   0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  0   0   1   0   0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
□:  0   0   1   0   1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY):
□:  1

ACCOMPLISHMENT LEVEL 4
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
□:  4
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
□:  1   0   0   0   0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  0   0   0   1   1
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  0   0   0   1   1
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
□:  1   1   1   1   1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
□:  2
TRAJECTORY SET 2
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
□:  1   0   0   0   0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  0   0   0   1   1
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  1   1   1   0   0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
□:  0   0   1   0   1

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
□:  2
TRAJECTORY SET 3
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
□:  1   0   0   0   0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
□:  1   1   1   0   0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

3.2.3. The Degrading Processor Model of the Third Semi-Annual Status Report

```
⎕:  0  0  0  1  1
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  1  0  1
ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕:  2
TRAJECTORY SET 4
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  1  0  0  0  0
PHASE 1:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  1  0  0
PHASE 2:
ENTER THE G DIAGONAL (SPACE BETWEEN EACH ENTRY):
⎕:  1  1  0  0  0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY):
⎕:  0  0  0  1  1
ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  (SPACE BETWEEN EACH ENTRY)
⎕:  1

PERFORMABILITY FOR_THIS MISSION ← 0.99999994  4.527068935E¯8  1.47116895E¯12
      1.471175086E¯8  2.023989308E¯12
```

Figure 25. A Sample METAPHOR Session to Evaluate the example of Section 3.2.3.-- continued)

4. Descriptions of METAPHOR Commands

In the following sections is presented a detailed description of each operational *METAPHOR* command. A *METAPHOR* session is also given for each command to illustrate the command's use. In addition, many of the other figures in this report present other examples of each command's use.

4.1. ALTER

The *ALTER* command is used to change the values of model parameters. Only those parameters which have been input to *METAPHOR* can be *ALTER*ed; if the parameter is undefined, no change will be allowed. *METAPHOR* responds to *ALTER* by printing

PUT AN X BELOW EACH ITEM TO BE CHANGED. HELP AVAILABLE.
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS

The abbreviations are as follows:

—*P*	The intraphase state transition (P) matrices
H	The interphase state transition (H) matrices
CONST.BAS.VARS	The number of constant basic variables and their associated probabilities
ALL.ACC.LEVELS	Using the present H and P matrices and the present constant basic variable information, determine the performability of the system. *METAPHOR* will ask for the appropriate information regarding the accomplishment levels.
PRESENT.ACC.LEVEL	Alter only the accomplishment level presently under consideration.
I	Initial vector
G	Characteristic matrices

F Characteristic vector (at present, this *ALTER* operation is not executable.)

V Vector characterizing the constant basic variables

NUM.TRAJ.SETS Alter the number of trajectory sets describing the accomplishment level under consideration

For each variable under which an X is typed, *METAPHOR* performs up to two actions:

1) Checks to insure the parameter has been defined, i.e., that the user has at some time specified a value for that parameter, and

2) If the parameter has been defined, asks the user for a new value.

If an item is undefined when an alteration is requested, an error message will be printed and that alteration suppressed. More than one item may be changed with a single alter command. Example:

```
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS
X X     ——         X X
```

This informs *METAPHOR* that the P and H matrices are to be changed and that all accomplishment levels are to be changed (i.e., the performability is to be calculated). To change the number of phases or associated states, type *EXIT* and begin *METAPHOR* again. *ALTER* is designed for correcting input errors, and in Version 1, is useful only in tandem with the *EVAL* command. An illustration of the use of the *ALTER* command is given in Figure 26.

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE ALTER COMMAND.
*** IT IS BASED ON THE SESSION IN FIGURE 13.
*** FIRST, TRY ALTERING EVERYTHING TO SEE WHAT HAPPENS.
***
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.
P H CONST.BAS.VARS ALL.ACC.LEVELS PRESENT.ACC.LEVEL I G F V NUM.TRAJ.SETS
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
P MATRICES ARE NOT DEFINED AT THIS TIME.
H MATRICES ARE NOT DEFINED AT THIS TIME.
CONSTANT BASIC VARIABLES ARE NOT DEFINED AT THIS TIME.
THE ACCOMPLISHMENT LEVELS ARE NOT DEFINED AT THIS TIME.
AN ACCOMPLISHMENT LEVEL IS NOT DEFINED AT THIS TIME.
I VECTOR IS NOT DEFINED AT THIS TIME.
G MATRICES ARE NOT DEFINED AT THIS TIME.
F VECTOR IS NOT DEFINED AT THIS TIME.
THE CONSTANT BASIC VARIABLE VECTOR IS NOT DEFINED AT THIS TIME.
THE NUMBER OF TRAJECTORY SETS IS NOT DEFINED AT THIS TIME.
⎕:  COM
*** NOW CONSIDER ALTER IN CONJUNCTION WITH THE EVAL COMMAND
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  2

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  DEDFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001

PHASE 2:

WHAT TYPE OF P MATRIX?
```

Figure 26. METAPHOR session illustrating the ALTER command

```
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  3

SPECIFY THE H MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1¯2:

        ETC.
         .
         .
         .
```

Figure 26. METAPHOR session illustrating the ALTER command -- continued)

4.2. BRIEF

BRIEF ON suppresses all output from *METAPHOR*, with the single exception of the final performability result and any accompanying warning. Prompts for inputs are still printed. *BRIEF ON* is most useful when the user has established a command file (see Section 2.9.) and wishes to reduce the amount of printing done (and hence reduce the real time required to enter the model and perform the computations). *BRIEF OFF* restores the printing of all *METAPHOR* output. The default is *BRIEF OFF*. Figure 27 gives an example of the use of the *BRIEF* command.

4.3. CALC

When the *CALC* command is issued, *METAPHOR* will evaluate any valid *APL* arithmetical expression input. Variables can be defined and utilized--however the user is advised to begin each variable name with the letter *U* (for "user") so as to prevent possible conflicts with variables internal to *METAPHOR*.

When *METAPHOR* is expecting an expression to evaluate, it begins the line with the prompt symbol *?*. To leave CALC mode, issue the command *EXIT*.

CALC is useful in two instances. First, the analyst may occasionally wish to perform a series of calculations such as adding or multiplying a sequence of numbers; second, the user may wish to assign often used or difficult to type values to variables, and then use the variables later as responses to queries from *METAPHOR*. (See Section 2.2.) The conversation in Figure 28 illustrates the use of *CALC*.

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION ILLUSTRATES THE USE OF THE BRIEF COMMAND
*** IT IS BASED ON THE SESSION IN FIGURE 5
*** DEALING WITH THE TMR SYSTEM OF SECTION 3.1.1.
***
*** FIRST, WE∆LL TURN OFF THE OUTPUT FROM METAPHOR
***
⎕:  BRIEF ON
BRIEF BRIEF ON
⎕:  EVAL
⎕:  1
⎕:   4
⎕:  NFAIL
⎕:  10
⎕:  .0001
⎕:  1
⎕:  3
⎕:   0
⎕:  2
⎕:  1 0 0 0
⎕:  1 1 0 0
⎕:  1
⎕:  1 0 0 0
⎕:  0 0 1 1 .


PERFORMABILITY FOR THIS MISSION ← 0.999997005  2.995004747E¯6
⎕:  COM
*** NOW LET'S TURN THE OUTPUT BACK ON
***
⎕: BRIEF OFF
BRIEF BRIEF OFF
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  EVAL

NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:
```

Figure 27. METAPHOR session illustrating the BRIEF command

```
WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
```

Figure 27. METAPHOR session illustrating the BRIEF command

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1

TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE USE OF THE CALC COMMAND
***
*** NOTE THAT COMMANDS OTHER THAN ΔEXITΔ WILL NOT
*** WORK IN CALC MODE
***
*** NOTE ALSO THE USER DEFINED VARIABLES
***
⎕:  EVAL

NUMBER OF PHASES?
⎕:  CALC
?
⎕:  EVAL
EVAL
?
⎕:  CALC
CALC
?
⎕:  ALTER
ALTER
?
⎕:  HELP
HELP
?
⎕:  ÷*1
0.3678794412
?
⎕:  UPI←○1
3.141592654
?
⎕:  EXIT
EXIT
NUMBER OF PHASES?
⎕:  EXIT
     EVAL
EVAL
     HELP
HELP
     ) SI
```

Figure 28. METAPHOR session illustrating the CALC command

4.4. COM

Occasionally, the analyst may wish to insert into his output some notes or comments concerning the status of the evaluation or modeling process. Also, if the user is employing a command file (see Section 2.9.), he may wish to document portions of the file. Such documentation is conveniently done by issuing the *COM* command and typing the comments on the following lines (after the prompt symbol ***). *METAPHOR* does no processing of the information given in response to *COM*. Prompts are given until a null line is entered (i.e., a carriage return with no preceding characters on the line), at which time *METAPHOR* leaves COM mode and returns to its state previous to the *COM* command. (See Section 2.7.) The *COM* command is illustrated in Figure 29. Also, practically every other session used as an illustration in this report makes extensive use of the COM command.

4.5. DATA

The *DATA* command is used to inspect the values of model parameters. Only those parameters which have been specified to *METAPHOR* can be viewed; if the parameter is undefined, the variable will not be displayed. *METAPHOR* responds to *DATA* by printing

PUT AN X BELOW EACH ITEM TO BE DISPLAYED. HELP AVAILABLE.
NUM.PHASES NUM.STATES P H NUM.CONST.BAS.VARS PROB.CONST.BAS.VARS

NUM.ACC.LEVELS NUM.TRAJ.SETS I G F V PERF

The abbreviations are as follows:

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS IS A METAPHOR COMMENT
*** ANY TEXT STRING CAN BE PRINTED¯¯NONE WILL BE EVALUATED.
*** HELP
*** EXIT
*** METAPHOR
*** EVAL
*** ABCDEFGHIJKLMNOPQ
*** RSTUVWXYZ
*** α⊥∩⌊∊_∇∆⍳∘'⎕|⊤○*?
*** ρ⌈~↓∪ω⊃↑⊂12345678
*** 90+× ←[]{,./¨≠<≤
*** ≥>¯( )  ÷ →⊣()};:\
***
***
*** EXIT THE COM MODE BY ENTERING A NULL LINE,
*** I.E., HITTING CARRIAGE RETURN WITHOUT TYPING
*** ANY CHARACTERS
***
```

Figure 29. METAPHOR session illustrating the COM command

NUM.PHASES	The number of phases
NUM.STATES	The number of states
P	The intraphase transition (P) matrices
H	The interphase transition (H) matrices
NUM.CONST.BAS.VARS	The number of constant basic variables
PROB.CONST.BAS.VARS	The probabilities of each of the constant basic variables
NUM.ACC.LEVELS	The number of accomplishment levels
NUM.TRAJ.SETS	The number of trajectory sets associated with the accomplishment level under consideration
I	The initial vector for the trajectory set under consideration
G	The characteristic matrices for the trajectory set under consideration
F	The characteristic vector for the trajectory set under consideration
V	The vector characterizing the constant basic variables for the trajectory set under consideration
PERF	The performability

For each variable under which an *X* is typed, *METAPHOR* performs up to two actions:

1) Checks to insure the parameter has been defined, i.e., that the user has at some time specified a value for that parameter, and

2) If the parameter has been defined, prints it.

If an item is undefined when a display is requested, an error message will be printed and that display will be suppressed. More than one item may be displayed with a single data command. Example:

```
NUM.PHASES NUM.STATES P H NUM.CONST.BAS.VARS PROB.CONST.BAS.VARS
   X          X                                 X
NUM.ACC.LEVELS NUM.TRAJ.SETS I G F V PERF
    X              X             X
```

This informs *METAPHOR* that the number of phases, states, and accomplishment levels as well as the probabilities of the constant basic variables and the performability are to be displayed. *DATA* is useful in Version 1 only in tandem with the *EVAL* command since the only parameters that can be displayed are those used in the scope of the *EVAL* command. An illustration of the use of *DATA* is presented in Figure 30.

4.6. ECHO

ECHO ON causes all input to *METAPHOR* to be repeated on the terminal. *ECHO ON* is helpful when either

1) The user has established a command file (see Section 2.9.) and wished to have a record printed of the session, or

2) The user is utilizing several user-defined variables as constants to be employed when inputting numerical answers (see Section 4.3.) and the user wishes to get confirmation of the value of those constants.

ECHO OFF suppresses all printing of the input. The default is *ECHO OFF*. In Figure 31 is presented an example of the use of *ECHO*.

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
             VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE DATA COMMAND
***
□:  EVAL

NUMBER OF PHASES?
□:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  3

NUMBER OF CONSTANT BASIC VARIABLES?
□:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
□:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
□:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
□:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
□:  1  1  0  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
□:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
```

Figure 30. METAPHOR session illustrating the DATA command

```
⎕:  1  0  0  0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

NUMBER OF PHASES IS 1
NUMBER OF STATES PER PHASE IS 4
THE P MATRICES ARE:

  9.970044955E¯1   2.992509492E¯3   2.994006246E¯6   9.985012493E¯10
  0.000000000E0    9.980019987E¯1   1.997002332E¯3   9.990005831E¯7
  0.000000000E0    0.000000000E0    9.990004998E¯1   9.995001666E¯4
  0.000000000E0    0.000000000E0    0.000000000E0    1.000000000E0
H MATRICES HAVE NOT BEEN DEFINED
THE NUMBER OF BASIC VARIABLES HAS NOT BEEN DEFINED
THE CONSTANT BASIC VARIABLES HAVE NOT BEEN DEFINED
THE NUMBER OF ACCOMPLISHMENT LEVELS IS 2
THE NUMBER OF TRAJECTORY SETS IS:
1
THE INITIAL VECTOR IS 1  0  0  0
G MATRICES NOT DEFINED
F VECTOR NOT DEFINED
THE CONSTANT BASIC VARIABLE VECTOR NOT DEFINED
THE PERFORMABILITY IS 0.999997005
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY
       ) :
⎕:  0  0  1  1


PERFORMABILITY FOR THIS MISSION ← 0.999997005  2.995004747E¯6
```

Figure 30. METAPHOR session illustrating the DATA command -- continued)

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
□:  COM
*** THIS SESSION DEMONSTRATES THE ECHO COMMAND.
*** IS IS BASED ON THE SESSION IN FIGURE 6,
*** DISCUSSED IN SECTION 3.1.2.
***
*** FIRST, LET US GO FOR A WHILE WITH THE ECHO TURNED OFF ( THE DEFAULT)
***
□:  EVAL

NUMBER OF PHASES?
□:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
□:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
□:  NFAIL
ENTER PHASE LENGTH
□:  10
ENTER COMPONENT FAILURE RATE
□:  0.0001
ENTER NUMBER OF GROUPS
□:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
□:  3

NUMBER OF CONSTANT BASIC VARIABLES?
□:  0

NUMBER OF ACCOMPLISHMENT LEVELS?
□:  2

ACCOMPLISHMENT LEVEL 0
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
□:  COM
*** NOW LETS CALCULATE THE INITIAL STATE PROBABILITIES
*** 1
□:  CALC
?
□:  UP←.99
0.99
```

Figure 31. METAPHOR session illustrating the ECHO command

```
?
⎕:  UQ←.01
0.01
?
⎕:  U3←UP*3
0.970299
?
⎕:  U2←3×UQ×UP*2
0.029403
?
⎕:  U1←3.N×UP.N×UQ.N*2
0.000297
?
⎕:  U0←UQ*3
1E¯6
?
⎕:  EXIT
EXIT
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  COM
*** LET∆S TURN THE ECHO ON...THIS WILL ALLOW US TO INSPECT THE
*** USER DEFINED VARIABLES AS WE USE THEM
***
⎕:  ECHO ON
⎕:  ECHO ON
ECHO ECHO ON
⎕:  1
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  UI←U3,U2,U1,U0
⎕:  0.970299  0.029403  0.000297  1E¯6
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0  0
⎕:  1  1  0  0

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  1
⎕:  1
TRAJECTORY SET 1
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  UI
⎕:  0.970299  0.029403  0.000297  1E¯6
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0  0  1  1
⎕:  0  0  1  1


PERFORMABILITY FOR THIS MISSION ← 0.9996403467  0.0003596532833
```

Figure 31. METAPHOR session illustrating the ECHO command -- continued)

4.7. EVAL

EVAL initiates a sequence of queries aimed at the computation of trajectory set probabilities, i.e., step 5) of the major steps outlined in the Introduction. The algorithm employed is based on the theory described in the third Semi-Annual Status Report ([4], Section 3.4).

The procedure is as follows. First, *METAPHOR* obtains from the user the number of phases, the number of states per phase, the intraphase (P) and interphase (H) matrices, the number of time-invariant basic variables, the probabilities of those time-invariant basic variables, and the number of accomplishment levels. Next *METAPHOR* computes the probability of each accomplishment level separately by obtaining the number of trajectory sets in the accomplishment level and then determining the probability of each trajectory set. This is done by procuring the initial state (I) vector, the characteristic (G) matrices, and the characteristic (F) vector (see Section 3.4 of [4] and [10]). Also, a characterization of the time-invariant basic variables is obtained. This latter characterization is called the "constant basic variable vector" ("V vector") in *METAPHOR* and is similar to the I vector. That is, each time-invariant variable k is associated with an entry in the constant basic variable (say V[k]), and whether the occurrence of that basic variable is allowed for a given trajectory set is stored in V[K]. Also, there is a column vector BASICVARIABLES defined such that the probability of the event corresponding to basic variable k is placed in BASICVARIABLES[k]. (See Section 2.4.) From these characterizations, the trajectory set probability is calculated according to

Probability = I * P(1) * G(1) * H(1) * P(2) * G(2) * H(2) * ... * H(n) * F * V * BASICVARIABLES

where n is the number of phases and * denotes matrix multiplication.

As an example of the use of *EVAL*, consider any of the sessions in Section 3.

4.8. EXIT

The command *EXIT* has two functions depending on the mode in which the command is issued. When in calc mode, *EXIT* returns *METAPHOR* to the previous mode in which *METAPHOR* was operating. When in COMMAND or COMMAND/REPLY mode, *METAPHOR* quits; all variable information is preserved, however program state information is lost and so the program cannot be continued. The host *APL* program returns to *APL* command mode. Figure 32 demonstrates the *EXIT* command.

4.9. HELP

Occasionally, when *METAPHOR* is in COMMAND or COMMAND/REPLY mode, the user may wish to know either his possible next actions or else further information concerning the question *METAPHOR* has asked. In such cases, the user can issue the *HELP* command and *METAPHOR* will respond with a short paragraph describing

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


⎕:  COM
*** THIS SESSION DEMONSTRATES THE EXIT COMMAND
***
*** FIRST, EXIT CAN BE EMPLOYED IN THE CALC MODE
***
⎕:  CALC
?
⎕:
?
⎕:  EXIT
⎕:  COM
*** SECOND, EXIT CAN BE USED TO LEAVE METAPHOR
***
⎕:  EXIT


    UCOMMENT←'NOW, WE ARE IN APL COMMAND MODE'
    EVAL
EVAL

    HELP
HELP

    UCOMMENT←'NONE OF THE METAPHOR COMMANDS WILL WORK HERE'
```

Figure 32. METAPHOR session illustrating the EXIT command

1) The various *METAPHOR* commands if in COMMAND mode, or

2) The specific model parameter with which the preceding question is concerned if in COMMAND/REPLY mode.

The *HELP* command has been included in *METAPHOR* to help it serve as a performability tutor; see Section 2.8. The *HELP* command is illustrated in Figure 33.

5. Description of Transition Matrix Types

When the user desires to enter a transition matrix, e.g., when specifying the H and P matrices, four types of entry methods are available: *DEDFAIL*, *GIVEN*, *IDENTITY*, and *NFAIL*. *GIVEN* is used when the user wishes to enter the matrix entirely, *DEDFAIL* and *IDENTITY* generate matrices after obtaining some parameter information from the user, while *IDENTITY* generates an identity matrix automatically. The subsections that follow describe each of these commands in detail.

5.1. DEDFAIL

The *DEDFAIL* algorithm computes transition matrices for special types of systems, assuming that the structure of the system is described in terms of "subsystems" where the state of each subsystem is either "operational" or "failed." Also, *DEDFAIL* assumes that all subsystems are alike and fail independently with the same constant failure rate. Finally, subsystems are assumed to fail permanently, i.e., once a subsystem has failed, it remains failed for the duration of the phase. *DEDFAIL* keeps track of each subsystem in the system, i.e., whether a given subsystem is operational or failed can be

```
MICHIGAN EVALUATION AID FOR PERPHORMABILITY
              VERSION 1


TYPE HELP FOR ASSISTANCE
⎕:  COM
*** THIS SESSION DEMONSTRATES THE HELP COMMAND.
***
⎕:  HELP
METAPHOR IS AN INTERACTIVE SOFTWARE PACKAGE AIDING THE MODELING
AND ANALYSIS OF PERFORMABILITY.  AT PRESENT, METAPHOR IS CAPABLE
ONLY OF EVALUATING CERTAIN PERFORMABILITY MODELS.
THE COMMANDS PRESENTLY AVAILABLE ARE:  EVAL,HELP, DATA, ALTER, CALC,
          COM, BRIEF [ON|OFF], ECHO [ON|OFF], AND EXIT.
DO YOU WANT MORE HELP?
⎕:
YES
THE COMMANDS CAN BE ENTERED AT ANY TIME EXCEPT IN RESPONSE TO A
YES/NO QUESTION.  THE COMMANDS ARE AS FOLLOWS:

          EVAL     EVALUATE A USER SUPPLIED PERFORMABILITY MODEL

          HELP     GIVE MORE INFORMATION ABOUT THE QUESTION BEING
                   ASKED

          DATA     DISPLAY VARIABLE INFORMATION AND MODEL PARAMETERS

         ALTER     CHANGE VARIABLE INFORMATION AND MODEL PARAMETERS

          CALC     ENTER THE APL CALCULATOR MODE.  TYPE 'EXIT' TO LEAVE.

           COM     ENTER COMMENTS ON THE OUTPUT

BRIEF [ON|OFF]     TURN BRIEF OUTPUT ON OR OFF

 ECHO [ON|OFF]     TURN INPUT ECHO ON OR OFF

          EXIT     LEAVE METAPHOR

DO YOU WANT REFERENCES?
⎕:
YES
FOR FURTHER INFORMATION ON PERFORMABILITY MODELING AND ANALYSIS, SEE

   J. F. MEYER, 'MODELS AND TECHNIQUES FOR EVALUATING THE
   EFFECTIVENESS OF AIRCRAFT COMPUTING SYSTEMS,' NASA GRANT
   NSG 1306, STATUS REPORT NO. 3, NOVEMBER 1977.

FOR FURTHER INFORMATION REGARDING METAPHOR, SEE

   J. F. MEYER, 'MODELS AND TECHNIQUES FOR EVALUATING THE
   EFFECTIVENESS OF AIRCRAFT COMPUTING SYSTEMS,' NASA GRANT
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
   NSG 1306, STATUS REPORT NO. 4, JULY 1978.

FOR FURTHER INFORMATION REGARDING APL, SEE

   S. PAKIN, 'APL\360 REFERENCE MANUAL,' SCIENCE RESEARCH
   ASSOCIATES,INC., CHICAGO, 1972.

⎕:  EVAL

NUMBER OF PHASES?
⎕:  HELP
ENTER THE NUMBER OF PHASES IN THE FINITE PHASE MODEL AS A SINGLE
POSITIVE INTEGER.
EXAMPLE:
     3
THIS INDICATES TO METAPHOR THAT THE MODEL TO BE EVALUATED HAS 3 PHASES
DO YOU WANT REFERENCES?
⎕:
N
NUMBER OF PHASES?
⎕:  1

NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  HELP
ENTER THE NUMBER OF STATES FOR EACH PHASE IN THE FINITE PHASE MODEL.

TYPE A POSITIVE INTEGER FOR EACH PHASE, SEPARATING EACH WITH SPACES AND/OR COMM
      S.
THE NUMBER OF STATES MUST BE A POSITIVE INTEGER.
EXAMPLE:
        4   3,5
THIS INDICATES TO METAPHOR THAT THE FIRST PHASE HAS 4 STATES, THE SECOND
PHASE HAS 3 STATES, AND THE THIRD PHASE HAS 5 STATES.
METAPHOR CHECKS TO MAKE SURE THAT THE NUMBER OF GROUPS OF STATES
MATCHES THE NUMBER OF PHASES INPUT EARLIER.  AN ERROR MESSAGE WILL BE
PRINTED IF THEY DO NOT MATCH.
DO YOU WANT REFERENCES?
⎕:
N
NUMBER OF STATES PER PHASE?  ( SPACE BETWEEN EACH NUMBER)
⎕:  4


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  HELP
TYPE ONE OF:   GIVEN, DEDFAIL, NFAIL, IDENTITY
DO YOU WANT MORE HELP?
⎕:
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
Y
ENTER ONE OF THE FOLLOWING TYPES FOR THE STATE TRANSITION ( P )  MATRIX:

     GIVEN      YOU WILL INPUT A P MATRIX, ONE ROW AT A TIME.

     DEDFAIL    METAPHOR WILL GENERATE A P MATRIX ASSUMING THE MATRIX
                REPRESENTS A SYSTEM HAVING N COMPONENTS, EACH FAILING
                INDEPENDENTLY AND EACH DISTINGUISHABLE.  THE STATE
                OF THE SYSTEM IS THE STATE OF EACH OF THE SUBSYSTEMS.
                YOU WILL BE ASKED THE LENGTH OF THE PHASE AND THE FAILURE
                RATE OF THE COMPONENTS.

     NFAIL      METAPHOR WILL GENERATE A P MATRIX ASSUMING THE MATRIX
                REPRESENTS A SYSTEM HAVING M GROUPS OF K( M)  COMPONENTS EACH.
                THE COMPONENTS FAIL INDEPENDENTLY AND THE STATE OF THE
                SYSTEM IS THE NUMBER OF ACTIVE ( NONFAILED)  COMPONENTS
                IN EACH GROUP.  YOU WILL BE ASKED THE NUMBER OF GROUPS,
                THE NUMBER OF COMPONENTS IN EACH GROUP, THE LENGTH
                OF THE PHASE, AND THE FAILURE RATE OF THE SUBSYSTEMS.

     IDENTITY   METAPHOR WILL GENERATE A P MATRIX ASSUMING THE MATRIX
                REPRESENTS A SYSTEM IN WHICH THERE IS NO FAILURE, I.E.,
                NO CHANGES IN STATES ARE MADE.  THUS, METAPHOR GENERATES
                AN IDENTITY MATRIX.

DO YOU WANT REFERENCES?
⎕:
N
WHAT TYPE OF P MATRIX?
⎕:  GIVEN
ENTER THE MATRIX, 1 ROW AT A TIME

ROW 1:
⎕:  HELP
ENTER AN M×N ARRAY, ONE ROW AT A TIME.  EACH ENTRY MUST
BE BETWEEN 0 AND 1 INCLUSIVE AND THE ENTRIES OF EACH ROW MUST SUM
TO ONE.  ENTER EACH ROW AS A SERIES OF N NUMBERS WITH SPACES AND/OR
COMMAS BETWEEN EACH.
EXAMPLE:
     .25 0.5,.1 0.15
HERE, THE MATRIX HAS FOUR ENTRIES PER ROW.

DO YOU WANT REFERENCES?
⎕:
N
ROW 1:
⎕:  1   0   0   0
ROW 2:
⎕:  1   0   0   0
ROW 3:
⎕:  1   0   0   0
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
ROW 4:
⎕:  1  0  0  0

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.

P  H  CONST.BAS.VARS  ALL.ACC.LEVELS  PRESENT.ACC.LEVEL  I  G  F V NUM.TRAJ.SE
PLACE AN X BELOW EACH ENTRY IN THE GIVEN LIST WHICH YOU WANT TO
CHANGE.  THE ABBREVIATIONS ARE AS FOLLOWS:

                  P     THE INTRAPHASE STATE TRANSITION ( P )  MATRICES

                  H     THE INTERPHASE STATE TRANSITION ( H )  MATRICES

     CONST.BAS.VARS     THE NUMBER OF CONSTANT BASIC VARIABLES AND
                        THEIR ASSOCIATED PROBABILITIES

     ALL.ACC.LEVELS     USING THE PRESENT H AND P MATRICES AND THE
                        PRESENT CONSTANT BASIC VARIABLE INFORMATION,
                        DETERMINE THE PERFORMABILITY OF THE SYSTEM.
                        METAPHOR WILL ASK FOR THE APPROPRIATE INFORMATION
                        REGARDING THE ACCOMPLISHMENT LEVELS.

  PRESENT.ACC.LEVEL     ALTER ONLY THE ACCOMPLISHMENT LEVEL
                        PRESENTLY UNDER CONSIDERATION.

                  I     INITIAL VECTOR

                  G     CHARACTERISTIC MATRICES

                  F     CHARACTERISTIC VECTOR

                        ( AT PRESENT, THIS ALTER OPERATION IS NOT EXECUTABLE.)

                  V     VECTOR CHARACTERIZING THE CONSTANT BASIC VARIABLES

      NUM.TRAJ.SETS     ALTER THE NUMBER OF TRAJECTORY SETS DESCRIBING
                        THE ACCOMPLISHMENT LEVEL UNDER CONSIDERATION

IF AN ITEM IS UNDEFINED WHEN AN ALTERATION IS REQUESTED, AN ERROR
MESSAGE WILL BE PRINTED AND THAT ALTERATION SUPPRESSED.  MORE THAN
ONE ITEM MAY BE CHANGED WITH A SINGLE ALTER COMMAND.
EXAMPLE:


P  H  CONST.BAS.VARS  ALL.ACC.LEVELS  PRESENT.ACC.LEVEL  I  G  F V NUM.TRAJ.SE
X          X                 X


THIS INFORMS METAPHOR THAT THE P AND H MATRICES ARE TO BE CHANGED AND THAT THE
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
PERFORMABILITY IS TO BE CALCULATED.  IF YOU WISH TO CHANGE THE NUMBER OF PHASES
OR ASSOCIATED STATES, TYPE END AND BEGIN METAPHOR AGAIN

DO YOU WANT REFERENCES?
⎕:
N
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.

P  H  CONST.BAS.VARS  ALL.ACC.LEVELS  PRESENT.ACC.LEVEL  I  G  F V NUM.TRAJ.SET
X
ALTERING P


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  DEDFAIL
ENTER PHASE LENGTH
⎕:  HELP
METAPHOR WILL GENERATE A P MATRIX ASSUMING THE MATRIX REPRESENTS
A SYSTEM HAVING N SUBSYSTEMS, EACH FAILING INDEPENDENTLY AND EACH
DISTINGUISHABLE.  THE FAILURES ARE ALSO ASSUMED TO BE
POISSON, AND ONCE A SUBSYSTEM HAS FAILED, IT CANNOT
BECOME GOOD AGAIN.
THE STATE OF THE SYSTEM IS THE STATE OF EACH OF
THE SUBSYSTEMS.  THE NUMBER OF STATES DECLARED FOR THE PHASE MUST BE
A POWER OF TWO.  YOU WILL BE ASKED THE LENGTH OF THE PHASE[ ENTER A
SINGLE POSITIVE INTEGER.  NEXT YOU WILL BE PROMPTED FOR THE FAILURE
RATE OF THE SUBSYSTEMS.  AGAIN ENTER A SINGLE POSITIVE NUMBER.  IF

THIS NUMBER IS NOT BETWEEN 1E¯1 AND 1E¯10, YOU WILL BE ASKED FOR CONFIRMATION.

DO YOU WANT REFERENCES?
⎕:
N
ENTER PHASE LENGTH
⎕:  10
ENTER SUBSYSTEM FAILURE RATE
⎕:  0.0001
NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  DATA
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
ENTER AN X BELOW EACH ITEM IN THE GIVEN LIST WHICH YOU WANT TO
DISPLAY.  THE ABBREVIATIONS ARE AS FOLLOWS:

        NUM.PHASES     THE NUMBER OF PHASES

        NUM.STATES     THE NUMBER OF STATES
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
                    P     THE INTRAPHASE TRANSITION ( *4)  MATRICES

  NUM.CONST.BAS.VARS     THE NUMBER OF CONSTANT BASIC VARIABLES

 PROB.CONST.BAS.VARS     THE PROBABILITIES OF EACH OF THE CONSTANT
                         BASIC VARIABLES

      NUM.ACC.LEVELS     THE NUMBER OF ACCOMPLISHMENT LEVELS

       NUM.TRAJ.SETS     THE NUMBER OF TRAJECTORY SETS ASSOCIATED WITH
                         THE ACCOMPLISHMENT LEVEL UNDER CONSIDERATION

                    I     THE INITIAL VECTOR FOR THE TRAJECTORY SET
                         UNDER CONSIDERATION

                    G     THE CHARACTERISTIC MATRICES FOR THE TRAJECTORY
                         SET UNDER CONSIDERATION

                    F     THE CHARACTERISTIC VECTOR FOR THE TRAJECTORY
                         SET UNDER CONSIDERATION

                    V     THE VECTOR CHARACTERIZING THE CONSTANT
                         BASIC VARIABLES FOR THE TRAJECTORY SET
                         UNDER CONSIDERATION

                 PERF     THE PERFORMABILITY

IF AN ITEM IS UNDEFINED WHEN A DISPLAY IS REQUESTED, AN ERROR MESSAGE
WILL BE PRINTED AND THAT DISPLAY WILL BE SUPPRESSED.  MORE THAN ONE ITEM
MAY BE DISPLAYED WITH A SINGLE DATA COMMAND.
EXAMPLE:

NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
X           X                                     X
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF
  X             X                          X

THIS INFORMS METAPHOR THAT THE NUMBER OF PHASES, STATES, AND ACCOMPLISHMENT
LEVELS AS WELL AS THE PROBABILITIES OF THE CONSTANT BASIC VARIABLES
AND THE PERFORMABILITY ARE TO BE DISPLAYED.

DO YOU WANT REFERENCES?
⎕:
N
PUT AN X BELOW EACH ITEM TO BE DISPLAYED.  HELP AVAILABLE.
NUM.PHASES  NUM.STATES  P  H  NUM.CONST.BAS.VARS  PROB.CONST.BAS.VARS
NUM.ACC.LEVELS  NUM.TRAJ.SETS  I  G  F  V  PERF

NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  ALTER
PUT AN X BELOW EACH ITEM TO BE CHANGED.  HELP AVAILABLE.
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
P  H  CONST.BAS.VARS  ALL.ACC.LEVELS  PRESENT.ACC.LEVEL  I  G  F V NUM.TRAJ.SET,
ALTERING P


SPECIFY THE P MATRICES FOR EACH PHASE, 1 PHASE AT A TIME

PHASE 1:

WHAT TYPE OF P MATRIX?
⎕:  NFAIL
ENTER PHASE LENGTH
⎕:  HELP
METAPHOR WILL GENERATE A P MATRIX ASSUMING THE MATRIX REPRESENTS A
SYSTEM HAVING M GROUPS OF K( M)  SUBSYSTEMS EACH, WHERE K IS A
FUNCTION OF THE GROUP.  THE SUBSYSTEMS FAIL INDEPENDENTLY
AND ARE ASSUMED TO HAVE A POISSON DISTRIBUTION.  ALSO, ONCE
A SUBSYSTEM HAS FAILED, IT CANNOT BECOME GOOD AGAIN.
THE STATE OF THE SYSTEM IS THE NUMBER OF ACTIVE SUBSYSTEMS
IN EACH GROUP.  THE NUMBER OF STATES DECLARED
FOR THE PHASE MUST BE THE PRODUCT OF ;THE NUMBER OF COMPONENTS
IN EACH GROUP PLUS ONE:.  FOR EXAMPLE, IF THE SYSTEM HAS 3 GROUPS
CONTAINING RESPECTIVELY 2, 5, AND 7 COMPONENTS, THEN THE PHASE HAS
( 2+1) ×( 5+1) ×( 7+1) ←144 STATES.

YOU WILL BE ASKED THE LENGTH OF THE PHASE[ ENTER A SINGLE POSITIVE INTEGER.
NEXT YOU WILL BE PROMPTED FOR THE FAILURE RATE OF THE COMPONENTS.
AGAIN ENTER A SINGLE POSITIVE NUMBER.  IF THIS NUMBER IS NOT BETWEEN
1E¯1 AND 1E¯10, YOU WILL BE ASKED FOR CONFIRMATION.  YOU WILL THEN
BE ASKED THE NUMBER OF GROUPS[ ENTER THIS AS A SINGLE POSITIVE INTEGER.
FINALLY, METAPHOR WILL REQUEST THE NUMBER OF COMPONENTS IN EACH GROUP.
THIS SHOULD BE INPUT AS A ROW OF POSITIVE INTEGERS SEPARATED BY
SPACES OR COMMAS.

DO YOU WANT REFERENCES?
⎕:
N
ENTER PHASE LENGTH
⎕:  10
ENTER COMPONENT FAILURE RATE
⎕:  0.0001
ENTER NUMBER OF GROUPS
⎕:  1
ENTER NUMBER OF COMPONENTS PER GROUP  ( SPACE BETWEEN EACH NUMBER) :
⎕:  3
NUMBER OF CONSTANT BASIC VARIABLES?
⎕:  HELP
ENTER THE NUMBER OF BASIC VARIABLES WHOSE PROBABILITIES REMAIN CONSTANT

THROUGHOUT THE MISSION INTERVAL ( I.E., THE NUMBER OF CONSTANT BASIC VARIABLES.
THE NUMBER SHOULD BE A SINGLE POSITIVE INTEGER.
EXAMPLE:
      2
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

THIS INFORMS METAPHOR THAT TWO CONSTANT BASIC VARIABLES ARE CONSIDERED IN THE MODEL.

DO YOU WANT REFERENCES?
⎕:
N
NUMBER OF CONSTANT BASIC VARIABLES?
⎕: 1

PROBABILITIES OF EACH CONSTANT VARIABLE? (SPACE BETWEEN EACH NUMBER)
⎕: *HELP*

ENTER THE PROBABILITIES OF THE BASIC VARIABLES WHOSE PROBABILITIES REMAIN CONS
NT

THROUGHOUT THE MISSION INTERVAL (I.E., THE NUMBER OF CONSTANT BASIC VARIABLES
THE PROBABILITIES SHOULD BE ENTERED AS A ROW OF POSITIVE NUMBERS
BETWEEN ZERO AND ONE, INCLUSIVE. THE NUMBERS SHOULD BE
SEPARATED BY SPACES AND/OR COMMAS. THE ORDER OF THE NUMBERS SHOULD
CORRESPOND TO THE ORDER OF THE CONSTANT BASIC VARIABLE VECTORS
WHICH WILL BE ASKED FOR LATER.
EXAMPLE:
.2, 0.3 .4,0.1
THIS INFORMS METAPHOR THAT THE PROBABILITIES OF THE FOUR CONSTANT
BASIC VARIABLES ARE 0.2, 0.3, 0.4, *AND* 0.1 *RESPECTIVELY. THE NUMBER*
OF CONSTANT VARIABLES DECLARED EARLIER MUST HAVE BEEN FOUR OR AN
ERROR MESSAGE WILL RESULT.

DO YOU WANT REFERENCES?
⎕:
N
PROBABILITIES OF EACH CONSTANT VARIABLE? (SPACE BETWEEN EACH NUMBER)
⎕: 0.99

NUMBER OF ACCOMPLISHMENT LEVELS?
⎕: *HELP*

ENTER THE NUMBER OF ACCOMPLISHMENT LEVELS FOR THIS MODEL AS A SINGLE POSITIVE
TEGER.
EXAMPLE:
5

THIS INDICATES TO METAPHOR THAT THE MODEL IT IS EVALUATING HAS 5 *ACCOMPLISHMEN*
LEVELS.

DO YOU WANT REFERENCES?
⎕:
N
NUMBER OF ACCOMPLISHMENT LEVELS?
⎕: 2

ACCOMPLISHMENT LEVEL 0

Figure 33. METAPHOR session illustrating the HELP command -- continued)

NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕: *HELP*
ENTER THE NUMBER OF TRAJECTORY SETS ASSOCIATED WITH THIS ACCOMPLISHMENT LEVEL AS A SINGLE POSITIVE INTEGER.
EXAMPLE:

THIS INFORMS METAPHOR THAT THE ACCOMPLISHMENT LEVEL HAS FOUR TRAJECTORY SETS DESCRIBING IT.

DO YOU WANT REFERENCES?
⎕:
N
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕: 1
TRAJECTORY SET 1
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY) :
⎕: *HELP*
ENTER THE INITIAL PROBABILITY DISTRIBUTION FOR THIS TRAJECTORY.
TYPE A PROBABILITY BETWEEN ZERO AND ONE INCLUSIVE CORRESPONDING TO EACH STATEΔS INITIAL PROBABILITY. SEPARATE EACH NUMBER WITH SPACES AND/OR COMMAS.

THE ORDER OF THE ENTRIES SHOULD CORRESPOND TO THE ORDER OF THE INITIAL STATES. THE NUMBER OF ENTRIES SHOULD BE THE SAME AS THE NUMBER OF STATES IN THE FIRST PHASE MODEL.
EXAMPLE:
.3 0.5,.2
THIS INFORMS METAPHOR THAT FOR THE TRAJECTORY SET UNDER CONSIDERATION, THE PROBABILITY THE SYSTEM BEGINS IN THE FIRST STATE OF PHASE 1 *IS* 0.2, *FOR THE*

SECOND STATE, THE PROBABILITY IS 0.5, *AND FOR THE THIRD STATE, THE PROBABILITY IS* 0.2.

DO YOU WANT REFERENCES?
⎕:
N
ENTER THE I VECTOR (SPACE BETWEEN EACH ENTRY) :
⎕: 1 0 0 0
ENTER THE F VECTOR (SPACE BETWEEN EACH ENTRY) :
⎕: *HELP*

ENTER THE CHARACTERISTIC (F) VECTOR FOR THE TRAJECTORY SET UNDER CONSIDERATIO

EACH ENTRY SHOULD BE EITHER 0 *OR* 1 *AND SHOULD BE SEPARATED FROM THE OTHER ENTIR*
S

BY SPACES AND/OR COMMAS. THE NUMBER OF ENTRIES SHOULD BE THE SAME AS THE NUMBER OF STATES OF THE FINAL PHASE MODEL. ALSO THE ORDER OF THE ENTRIES SHOULD CORRESPOND TO THE ORDER OF THE STATES AS CONSIDERED ELSEWHERE IN THE METAPHOR PACKAGE FOR THE FINAL PHASE.
EXAMPLE:
1 0, 1 ,0

Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
THIS INFORMS METAPHOR THAT THE CHARACTERISTIC VECTOR FOR THIS TRAJECTORY
SET IS
   1
   0
   1
   0

DO YOU WANT REFERENCES?
⎕:
N
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1  1  0  0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY)
⎕:  HELP
ENTER A VECTOR OF 0'S, 1'S, AND 2'S TO INDICATE WHETHER THE
CONSTANT BASIC VARIABLE S OCCURRENCES OR NON- OCCURRENCES SHOULD BE
CONSIDERED IN THE TRAJECTORY SET.  THE CODING IS AS FOLLOWS:

       0       THE CORRESPONDING BASIC VARIABLE'S OCCURRENCE SHOULD
               BE CONSIDERED

       1       THE CORRESPONDING BASIC VARIABLE'S NON- OCCURRENCE
               SHOULD BE CONSIDERED

       2       EITHER THE OCCURRENCE OR  NON- OCCURRENCE OF THE
               CORRESPONDING BASIC VARIABLE SHOULD BE CONSIDERED
               ( I.E., THE BASIC VARIABLE IS A 'DON'T CARE')

ENTER A ROW OF 0'S, 1'S, AND 2'S, SEPARATING EACH ENTRY BY SPACES
AND/OR COMMAS.  THE NUMBER OF ENTRIES SHOULD BE THE SAME AS THE
NUMBER OF CONSTANT BASIC VARIABLES DECLARED EARLIER.  ALSO THE ORDER
OF THE ENTRIES SHOULD CORRESPOND TO THE ORDER OF THE BASIC VARIABLES
AS CONSIDERED ELSEWHERE IN THE METAPHOR PACKAGE.
EXAMPLE:
         0,1  1 , 2
THIS INDICATES TO METAPHOR THAT FOR THIS TRAJECTORY SET, THE OCCURRENCE
OF THE FIRST CONSTANT BASIC VARIABLE IS IMPORTANT TO THE TRAJECTORY
SET, THE NON- OCCURRENCE OF THE SECOND AND THIRD CONSTANT BASIC VARIABLES
IS IMPORTANT, AND THAT THE FOURTH CONSTANT BASIC VARIABLE IS IRRELEVANT.

DO YOU WANT REFERENCES?
⎕:
N

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY)
⎕:  1

ACCOMPLISHMENT LEVEL 1
NUMBER OF TRAJECTORY SETS FOR THIS ACCOMPLISHMENT LEVEL?
⎕:  2
TRAJECTORY SET 1
```


Figure 33. METAPHOR session illustrating the HELP command -- continued)

```
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1   0   0   0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1   1   0   0

ENTER THE 1 ELEMENT CONSTANT BASIC VARIABLE VECTOR  ( SPACE BETWEEN EACH ENTRY)
⎕:  0
TRAJECTORY SET 2
ENTER THE I VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  1   0   0   0
ENTER THE F VECTOR ( SPACE BETWEEN EACH ENTRY) :
⎕:  0   0   1   1
⎕:  0   0   1   1


PERFORMABILITY FOR THIS MISSION ← 0.999997005  2.995004747E¯6
```

Figure 33. METAPHOR session illustrating the HELP command -- continued)

deduced from the state of the system. In *METAPHOR*, the most important use of *DEDFAIL* is in modeling a system wherein each subsystem (e.g., processor) is dedicated to a different task (hence the name *DEDFAIL*). In such situations, the processing capability generally depends on the state of each subsystem and hence the system state must convey the state of each subsystem. When using the *DEDFAIL* command, the number of states declared for the phase must be a power of two. If the system has N subsystems, then the resulting transition matrix is NxN where the $(i,j)^{th}$ entry denotes the probability that the system is in state j at the end of the phase given it was in state i at the beginning of the phase. The i^{th} row or column of the matrix represents the state determined as follows: Assign each subsystem a unique integer between 1 and N. Take the binary representation of $(2^N)-1-i$. Then the i^{th} digit of the binary representation (read from left to right) represents the state of the corresponding subsystem in the system, 0 if failed, 1 if not failed. In other words, take the number of states n in the phase and write all the integers from 0 to n in binary form, inserting leading zeros if necessary to make all the representations $\log_2(n)$ long. Then given the representation of a state $a_{\log(n)}$ $a_{\log(n)-1}$ ... a_2 a_1, the digit a_i is 0 if subsystem i is operational, 1 otherwise. For the transition matrix, states are ordered by their numerical value in descending order. Hence the top row and first column correspond to the state where all subsystems are operational, while the bottom row and last column correspond to the state where all subsystems are failed. For instance, if the system has 16 states, then the rows and columns of the matrix would represent states in the order

```
1111
1110
1101
1100
1011
1010
1001
1000
0111
0110
0101
0100
0011
0010
0001
0000 .
```

As an example, consider a system with two subsystems having the transition in Figure 34. With a failure rate of 0.001 and a phase length of 10, the following transition matrix results:

	(1,1)	(1,0)	(0,1)	(0,0)
(1,1)	9.98E-1	9.99E-4	9.99E-4	9.99E-7
(1,0)	0.00E0	9.99E-1	0.00E0	1.00E-3
(1,0)	0.00E0	0.00E0	9.99E-1	1.00E-3
(0,0)	0.00E0	0.00E0	0.00E0	1.00E0

Here (1,1) means that both subsystems are not failed, (1,0) that the first subsystem is not failed but the second one is, (0,1) that the first subsystem is failed but the second is not, and (0,0) that both subsystems are failed.

DEDFAIL is equivalent to *NFAIL* when *NFAIL* has N groups of 1 subsystem each.

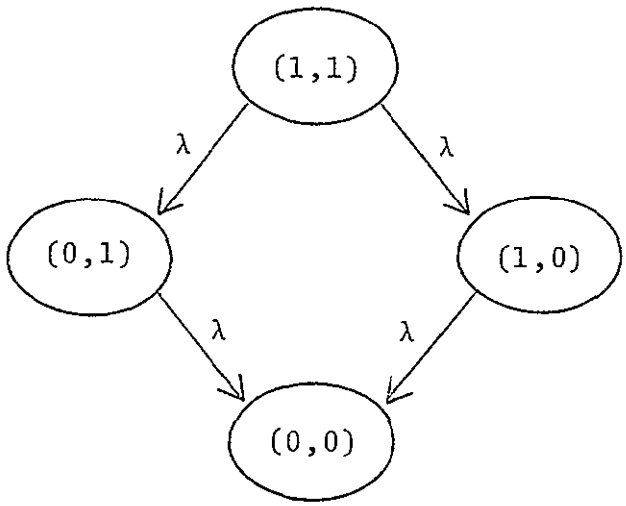


Figure 34. Transition graph for the system of Section 5.1.

5.2. GIVEN

The *GIVEN* transition matrix type allows the analyst to supply his own transition matrix. After the user specifies *GIVEN*, *METAPHOR* prompts the user to enter the matrix one row at a time. Each row is checked to insure that it sums to one--if not, the row must be reentered. *GIVEN* can be specified for either H or P matrices. Array type *GIVEN* is designed for transition matrices which either can not be generated by other *METAPHOR* array generating functions or are too expensive to repetitively generate.

5.3. IDENTITY

When *IDENTITY* is specified as a transition matrix type, *METAPHOR* automatically generates an identity matrix, i.e., a matrix with all 1's on the main diagonal. *IDENTITY* is intended for transitions involving no change in state; that is, the states of the system at the initiation and the termination of the transition period are same. Both P or H matrices can be specified with *IDENTITY*.

5.4. NFAIL

The *NFAIL* algorithm computes transition matrices for special types of systems, assuming that the structure of the system is described in terms of "subsystems" where the state of each subsystem is either "operational" or "failed." Also, *NFAIL* assumes that all subsystems are alike and fail independently with the same constant failure rate. Finally, subsystems are assumed to fail permanently, i.e., once a subsystem has failed, it remains failed for the duration of the phase.

NFAIL assumes that the subsystems of the system are lumped into groups. *NFAIL* then keeps track only of the number of subsystems which are operational within each of these groups. (Compare with the *DEDFAIL* command in Section 5.1.) For instance, if two tasks and four processors are configured such that two processors are executing each task, then failure of either processor assigned to a given task will have the same effect on system performance. Accordingly, processors sharing the same task can be lumped, resulting in 2 groups with 2 processors per group. The state of the system represents the number of active (non-failed) subsystems in each group. The number of states declared for the phase is determined as follows: to the number of subsystems in each group add 1 and take the product of the resulting terms. For example, consider a system having 3 groups containing respectively 5, 2, and 7 subsystems. the phase then has (5+1)*(2+1)*(7+1)=144 states. In general, if the number of states for the phase is N, then the resulting matrix will be NxN in shape, such that the $(i,j)^{th}$ entry will be the probability that the system is in state j at the end of the phase given it was in state i at the beginning of the phase.

Consider a system having M groups and suppose the number of subsystems in the m^{th} group is denoted by the function K(m). Then the i^{th} row or column of the resulting transition matrix represents the state determined as follows: First, take the mixed radix number system such that the j^{th} place (counted from the right) of a number has weight (multiplier):

$$\begin{cases} 1, & \text{if } j=1 \\ (K(1)+1)(K(2)+1)\ \ldots\ (K(j-1)+1), & \text{if } j>1 \end{cases} .$$

The values ("digits") that the j^{th} place can take on are 0, 1, ..., K(j). For instance, with the 5,2,7 system above, we would employ a number system having 3 places. The first place has weight 1 and can range form 0 to 7, the second place has weight 7+1=8 and can vary from 0 to 2, and the third place has weight (7+1)*(2+1)=24 and can range from 0 to 2. The base ten number 55 would then be written '207' since 55 = 2*24 + 0*8 + 7*1, while the base ten number 17 would be expressed '021' because 17 = 0*24 + 2*8 + 1*1. A number represented in the above system has the following interpretation: the value of each digit of a number denotes the state of the corresponding group, i.e., the number of operational subsystems in the group. Hence '207' means groups 1, 2, and 3 have 2, 0, and 7 active subsystems respectively. and '021' means groups 1, 2, and 3 have 1, 2, and 0 active subsystems respectively. An easy way to determine the states and their corresponding position in the transition matrix is as follows:

1) Take the number of subsystems n_i in each group i <u>in the order presented</u> (in response to the question "*ENTER NUMBER OF COMPONENTS PER GROUP*") and write the digits from 0 to n_i.

2) Then write all the numbers composed of these digits <u>in the order presented</u>.

These numbers, in descending order, represent the states corresponding to the rows and columns of the transition matrix.

5.4. NFAIL

For instance, with the 5,2,7 system, the rows and columns of the matrix would represent the 144 states in the order:

527
526
525
524
523
522
521
520
517
.
.
.
501
500
427
426
425
.
.
.
011
010
007
006
005
004
003
002
001
000 .

For example, consider a system with 2 groups, the first having 1 subsystem and the second 2 subsystems, having the transition diagram in Figure 35. with a transition rate of 0.001 and a period of 10, the following transition matrix results:

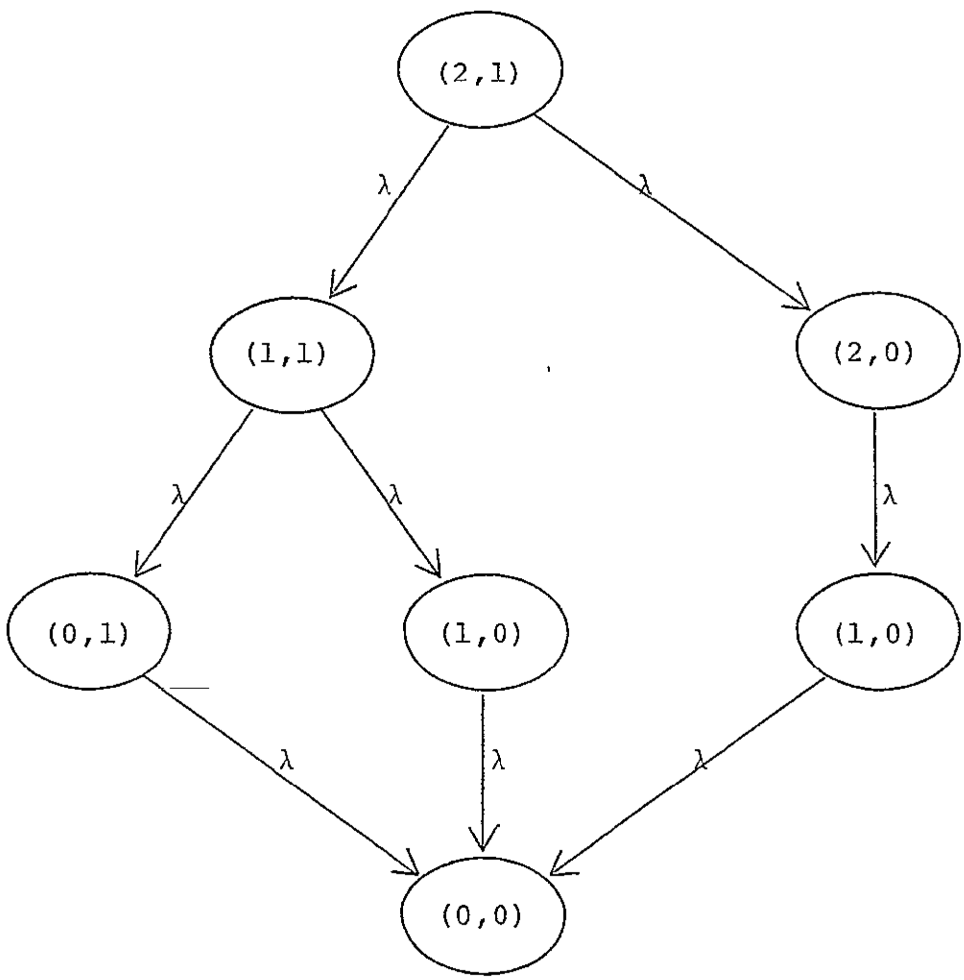


Figure 35. Transition graph for the system of Section 5.4.

5.4. NFAIL

	(2,1)	(2,0)	(1,1)	(1,0)	(0,1)	(0,0)
(2,1)	9.704E-1	9.753E-3	1.951E-2	1.960E-4	9.802E-5	9.851E-7
(2,0)	0.000E0	9.802E-1	0.000E0	1.970E-2	0.000E0	9.901E-5
(1,1)	0.000E0	0.000E0	9.802E-1	9.851E-3	9.851E-3	9.901E-5
(1,0)	0.000E0	0.000E0	0.000E0	9.900E-1	0.000E0	9.950E-3
(0,1)	0.000E0	0.000E0	0.000E0	0.000E0	9.900E-1	9.950E-3
(0,0)	0.000E0	0.000E0	0.000E0	0.000E0	0.000E0	1.000E0

where (2,1) means that group 1 has one operational subsystems group 2 has two operational subsystems, (2,0) means that group 1 no active subsystems and group 2 two active subsystems, and forth. *NFAIL* is equivalent to *DEDFAIL* when N groups of subsystem each are specified.

6. References

[1] D. G. Furchtgott (under the direction of J. F. Meyer), "METAPHOR (Version 1) Programmer's Guide," SEL Report No. 128, Systems Engineering Lab, The University of Michigan, Ann Arbor, MI, January 1979.

[2] J. F. Meyer, "Models and techniques for evaluating the effectiveness of aircraft computing systems," Semi-Annual Status Report Number 1, November, 1976.

[3] J. F. Meyer, "Models and techniques for evaluating the effectiveness of aircraft computing systems," Semi-Annual Status Report Number 2, July, 1977.

[4] J. F. Meyer, "Models and techniques for evaluating the effectiveness of aircraft computing systems," Semi-Annual Status Report Number 3, NASA Report CR158992, January 1978.

[5] J. F. Meyer, "Models and techniques for evaluating the effectiveness of aircraft computing systems," Semi-Annual Status Report Number 4, NASA Report CR158993, July 1978.

[6] J. F. Meyer, "A model hierarchy for evaluating the effectiveness of computing systems," Texte des Conferences III-e Congres National de Fiabilite, Tome II, Perros-Guirec, France, pp. 539-555, September 1976.

[7] R. A. Ballance and J. F. Meyer, "Functional dependence and its application to system evaluation," Proc. of the 1978 Johns Hopkins Conf. on Information Sciences and Systems, Baltimore, MD, pp. 280-285, March 1978.

[8] J. F. Meyer, "On evaluating the performability of degradable computing systems," Proc. 1978 Int'l Symp. on Fault-Tolerant Computing, Toulouse, France, pp. 44-49, June, 1978.

[9] D. G. Furchtgott and J. F. Meyer, "Performability evaluation of fault-tolerant multiprocessors," 1978 Government Microcircuit Applications Conference Digest of Papers, Monterey, California, pp. 362-365, November, 1978.

[10] L. T. Wu and J. F. Meyer, "Phased models for evaluating the performability of computing systems," Proc. of the 1979 Johns Hopkins Conf. on Information Sciences and Systems, Baltimore, MD, pp.426-431, March 1979.

[11] J. F. Meyer, D. G. Furchtgott, and L. T. Wu, "Performability evaluation of the SIFT computer", Proc. 1979 Int'l Symp. on Fault-Tolerant Computing, Madison, Wisconson, pp. 43-50, June, 1979.

[12] J. F. Meyer, "Performability Modeling with Continuous Accomplishment Sets," SEL Report No. 137, Systems Engineering Lab, The University of Michigan, Ann Arbor, MI, July 1979. January 1979.

[13] K. E. Iverson, A Programming Language. New York: Wiley, 1962.

[14] S. Pakin, APL/360 Reference Manual. Chicago, IL: Science Research Associates, 1972.

[15] P. L. Tison, "An Algebra for Logic Systems--Switching Circuits Application," IEEE Trans. Comput., vol. C-20, pp. 339-351, November, 1971.

[16] J. D. Esary and H. Ziehms, "Reliability of Phased Missions," Reliablity and Fault Tree Analysis, SIAM, Philadelphia, PA, pp. 213-236, 1975.

7. Index