

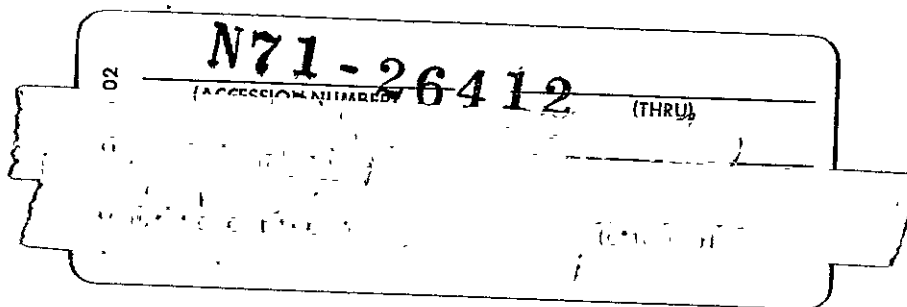
THE USE OF GERT IN PLANNING STRATEGIES
FOR DEVELOPMENT TYPE PROJECTS*

by

Joseph J. Moder, Principal Investigator
Ramon S. Gomez, Research Assistant, and
Richard A. Clark, Research Assistant

Department of Industrial Engineering
and Systems Analysis
School of Engineering
University of Miami
Miami, Florida

June 15, 1970



Research Sponsored

by

Electronics Research Center
National Aeronautics and Space Administration
NASA Contract NAS-12-2080

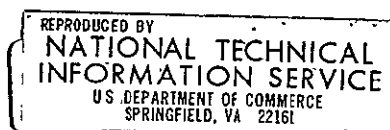


TABLE OF CONTENTS

	<u>Page</u>
1. Introduction and Summary	1
2. The Role of GERT in Management System Design and Documentation	3
3. The GERT-II Simulator	5
3.1 GERT-II Nomenclature	6
3.2 Modifications to GERT-II	8
3.3 Variance Reduction Techniques	12
4. Application of GERT to Development - Type Projects	13
GERT Network for the Development of a Munitions System	15
5. Summary	24
References	25
Appendix I - The GERT Program as Modified	
Appendix II - Variance Reduction in the GERT-II Simulator	
Appendix III - Procedure to Input Data	

NOTICE TO USERS

Portions of this document have been judged by the Clearinghouse to be of poor reproduction quality and not fully legible. However, in an effort to make as much information as possible available to the public, the Clearinghouse sells this document with the understanding that if the user is not satisfied, the document may be returned for refund.

If you return this document, please include this notice together with the IBM order card (label) to:

Clearinghouse
Attn: 152.12
Springfield, Va. 22151

THE USE OF GERT IN PLANNING STRATEGIES
FOR DEVELOPMENT TYPE PROJECTS*

by

Joseph J. Moder, Professor,
Ramon S. Gomez, Research Assistant, and
Richard A. Clark, Research Assistant

Department of Industrial Engineering and
Systems Analysis

University of Miami,
Miami, Florida

June 15, 1970

1. Introduction and Summary

The authors' philosophy regarding the applications of GERT is quite similar to that held by most people about PERT/CPM, which is a restricted subset of the logic embodied in GERT. That is, a great deal of benefit can be derived from drawing the network which represents a project plan, or a management policy regarding the operation of some system. However, using GERT nomenclature, the range of project types is enlarged considerably.

The hypothesis posed here is that the GERT network is an excellent means for documenting a project plan or a policy, and then examining it qualitatively for omissions, redundancies, inefficiencies, and inconsistencies. This concept will be illustrated below by reference to the network for a management policy governing the handling of university inventions.

*This study was supported by NASA Contract, NAS 12-2080, entitled, "The Use of Gert in Development Program Management."

The addition of estimates of time, cost, resource requirements, branching probabilities, and other parameters associated with the network activities then leads to outputs that permit a quantitative assessment of the plan or policy to be made. The latter ideas will be developed in detail using as a vehicle, a developmental type project typical of those found in fields such as automotive, electronic, and weapons systems development where the periodic development of "new models" frequently follows established patterns. GERT is ideally suited to model such projects which frequently embody milestone events of a special nature. For example, branching events occur where knowledge gained in preceding activities lead the project along only one of several possible branches, and in cases where the preceding activities are of a testing nature, one or more of these branches may recycle back to some prior project event.

In development projects one frequently resorts to parallel development strategies, and other risky procedures to "buy time."^{1*} In very simple networks the effects of alternative strategies can be estimated by inspection with an acceptable degree of accuracy. However, the presence of a number of branching events with feedback loops can quickly render this problem unmanageable from a "hand-computation" point of view. The use of the GERT-II simulation program will be illustrated for such a case.

Several ancillary topics will be presented along with the development of the above ideas. In particular, variance

*Numbered references are given at the end of this report.

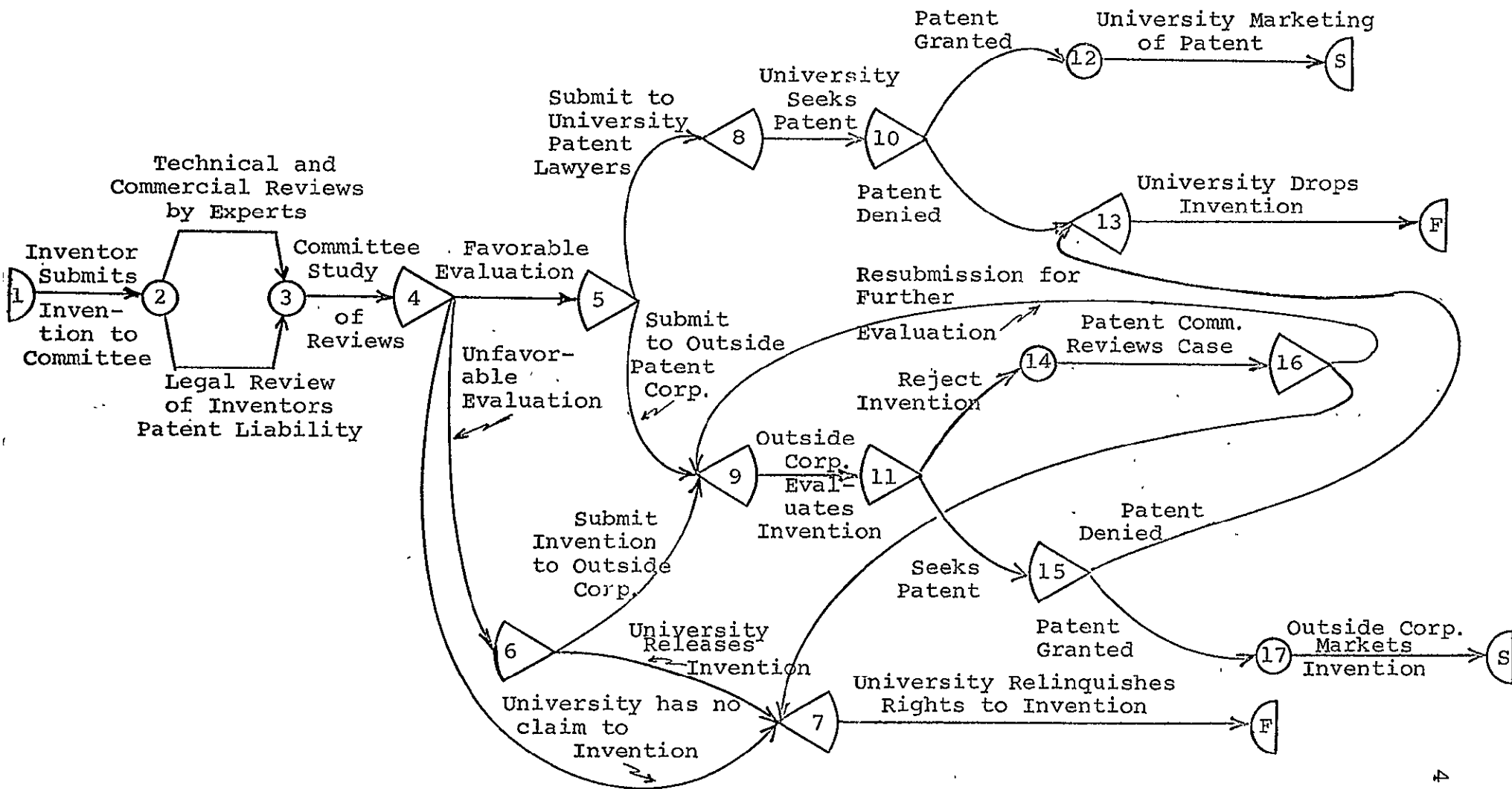
reduction techniques will be considered to reduce computer time requirements, and several modifications to the standard GERT-II simulator that were required to handle a development project network problem will be presented.

2. The Role of GERT in Management System Design and Documentation.

To illustrate the use of the GERT nomenclature in the documentation and qualitative study of a management policy, consider the network shown in Figure 1 which gives the operating policy and procedure for a typical University Patent Committee. This network has one initial event which occurs with the submission of an invention by the inventor to the Patent Committee. It has three terminal events which cover the possible modes of disposition of an invention. In developing this network, several aspects of the existing policy were found to need clarification and refinement. However, the resulting policy as stated in Figure 1 still contains certain omissions, inconsistencies, and inefficiencies.

The simultaneity of the two activities from node 2 to node 3 is inefficient since a negative legal review precludes the need for a technical review. Also, activity 4-7 represents a policy omission; it should be modified to give the inventor the option to have the university process his invention through their patenting and marketing channels. Finally, experience has shown that activity 6-7 represents an inconsistency in the policy. A person whose invention happens to get a poor evaluation from a patentability or marketability

Figure 1. POLICY FOR A UNIVERSITY PATENT COMMITTEE.



point of view is then given all rights to the invention, whereas a good evaluation of the same invention would give the inventor only partial royalty rights. This represents an important inconsistency because the risks of a good invention getting a poor review are not trivial.

The policy shown in Figure 1 does not appear to have redundant type errors. However, it has been the authors' experience that this type of error is quite prevalent in networks depicting information flow, notably in inventory and procurement type systems. Redundancy in record keeping has always been a problem in system design.

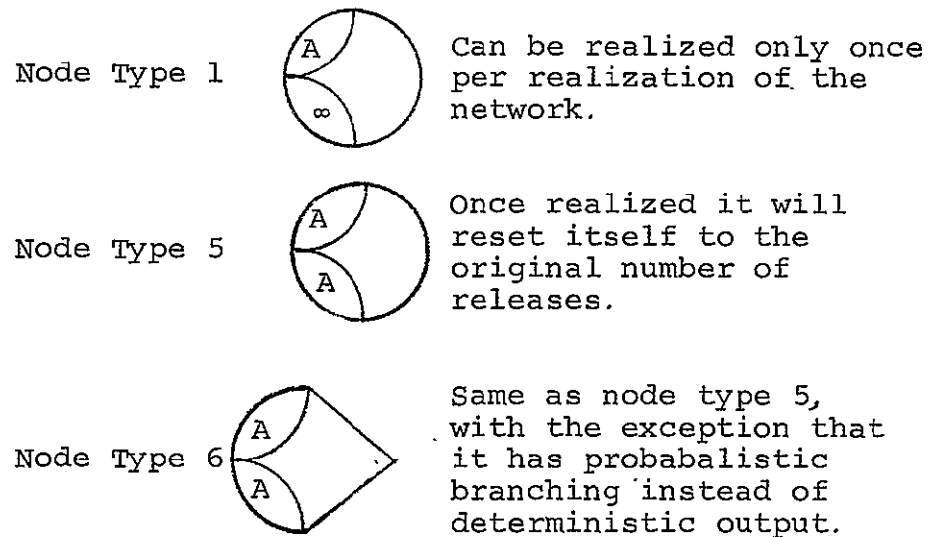
It is the authors' contention that while the problems described above could have been detected and possibly corrected without the GERT network, the precise expression of a complete policy or a plan in the form of a network cannot be duplicated by a written description. For this reason, GERT type nomenclature should be a standard tool of the systems analyst which has potential value in analysis and design, as well as communication.

3. The GERT-II Simulator

In this report, it will be assumed that the reader is familiar with the GERT-II Simulator as described in reference (2). A number of modifications and additions to this program were made to accomodate the type of problems being considered in this study. These modifications will be described in detail in this report. The modified program, called GERT-IIA, is given in Appendix I.

3.1 GERT-II Nomenclature

For the networks presented in this study, three node types from the original GERT-II are sufficient. They are shown in Figure 2. Node Type 1 is used for the start (source)



A = number of activity completions necessary to realize the node.

Figure 2. Three node types from GERT-II to Be Used in This Study.

and end (sink) nodes of the network, while the other two node types are used throughout the remainder of the network.

In GERT-II it is possible to change one node into another. This is done to permit changes in the output activities from a node, and it is brought about by the completion of a numbered activity. An example is given in Figure 3, along with illustrations of the three node types described above. In Figure 3 when branch 8 to 4 (activity number 2) occurs, node 5 is changed into node 10 to accommodate an altered sequence of subsequent activities on the recycle path. Similarly, when branch 12 to 3 (activity number 3)

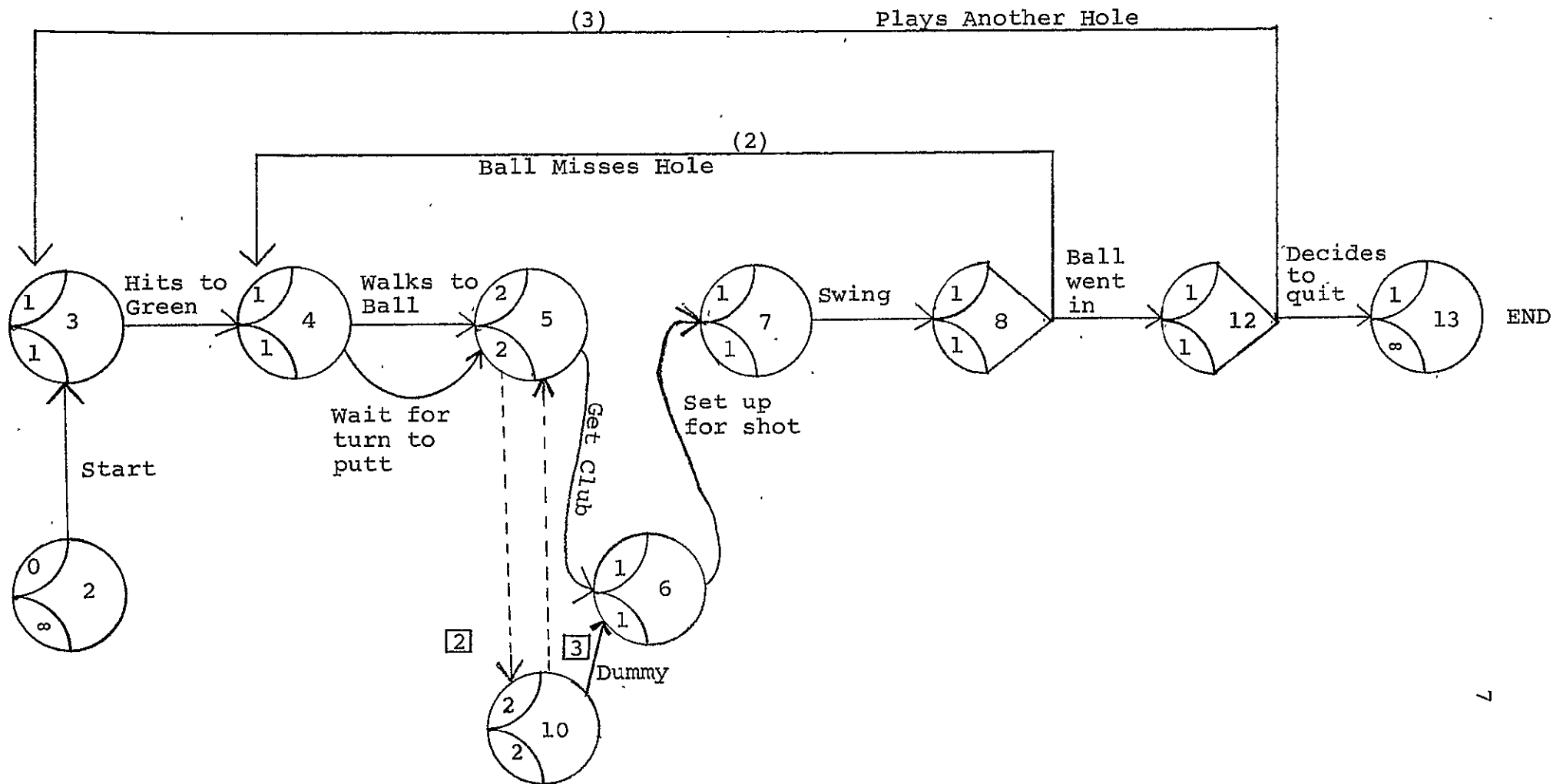


Figure 3. Illustration of Node Types and Node Changing - A Golfer on the Putting Greens.

occurs, node 10 is returned to the normal state, node 5, to accomodate the play of the next hole.

3.2 Modifications to GERT-II.

To accomplish the simulation of development-type projects, three techniques were developed and added to the GERT-II Simulator, i.e.

- 1) activity clear,
- 2) modification of node number of releases, and
- 3) logic node reset.

A cost collection routine was also added to test the overall effectiveness of alternate project management policies. The ability to consider resource constraints would be useful, but is not considered in this study.

Activity Clear and Modification of Node Number of Releases

In modeling development type projects, it is frequently necessary to abandon the work on some activity(s) because the results of some other activity is not successful. Thus it is necessary to destroy an activity(s) that may be in progress, or may already be completed and thus has partially realized a node by reducing its counter of releases to node realization.

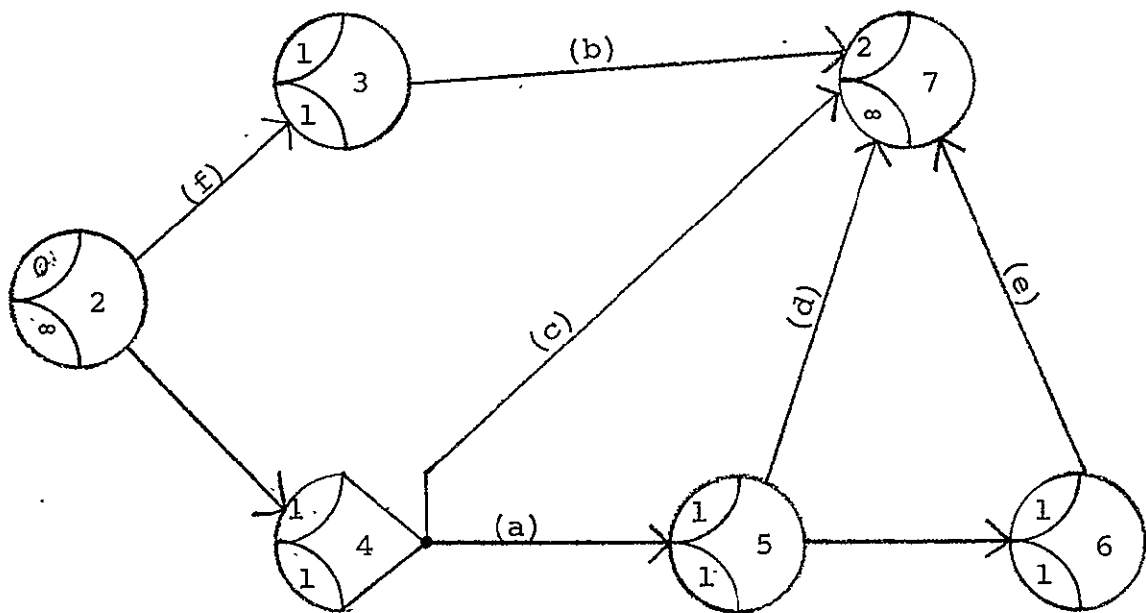


Figure 4. Example of Activity and Node Clear Procedures.

In the network shown in Figure 4, suppose we want node 7 to be realized by either the completion of activities (b) and (c), or activities (d) and (e). There is no problem if node 4 branches to activity (c); node 7 will eventually be realized by the completion of activities (b) and (c). However if node 4 branches to activity (a), then the activity clear routine is required. In this case, activity (b) may be in one of three states when activity (a) is completed.

- 1) Activity (b) can be completed, in which case the node 7 counter will have been changed to 1 activity to go before the node is realized. In this case the counter will be reset to 2 by the modification of node number of releases routine.
- 2) Activity (b) can be in progress, in which case it will be halted by the activity clear routine.

- 3) Activity (b) may not have been started, but it is free to be started as soon as activity (f) is completed. In this case, activity (f) will be halted by the activity clear routine.

Logic Node Reset

The reason for using the logic node reset is best explained by an example such as shown in Figure 5. If activity (6) is completed, node 9 must be reset to 2 or to 3, depending on whether node 5 branches to path III-A or III-B. The logic node, which is node number 1, is originally set to 3. If path III-B is taken, the logic node is set to 2 by the completion of activity (8). Then, on the completion of activity (6), node 9 is reset to the number of releases of the logic node. Node 9 will then be realized when paths I and II are completed. However, if path III-A is taken, the logic node setting remains at 3. Now, if activity (6) is realized, node 9 is reset to 3 since this case requires that paths I, II and III-A be reworked to realize node 9.*

Addition of Cost Parameters

To evaluate alternative policies for carrying out a project, GERT II-A includes an enumerative cost feature.

*The above result could have been accomplished by changing node 9 into another node requiring two activity completions for realization. However, it is faster in large networks to use the logic node, which eliminates a search of each node in the network. So that the data of GERT-II programs would remain compatible with GERT-II-A, node 1 was the only node used as a logic node, however, any number of nodes could be so designated in the program.

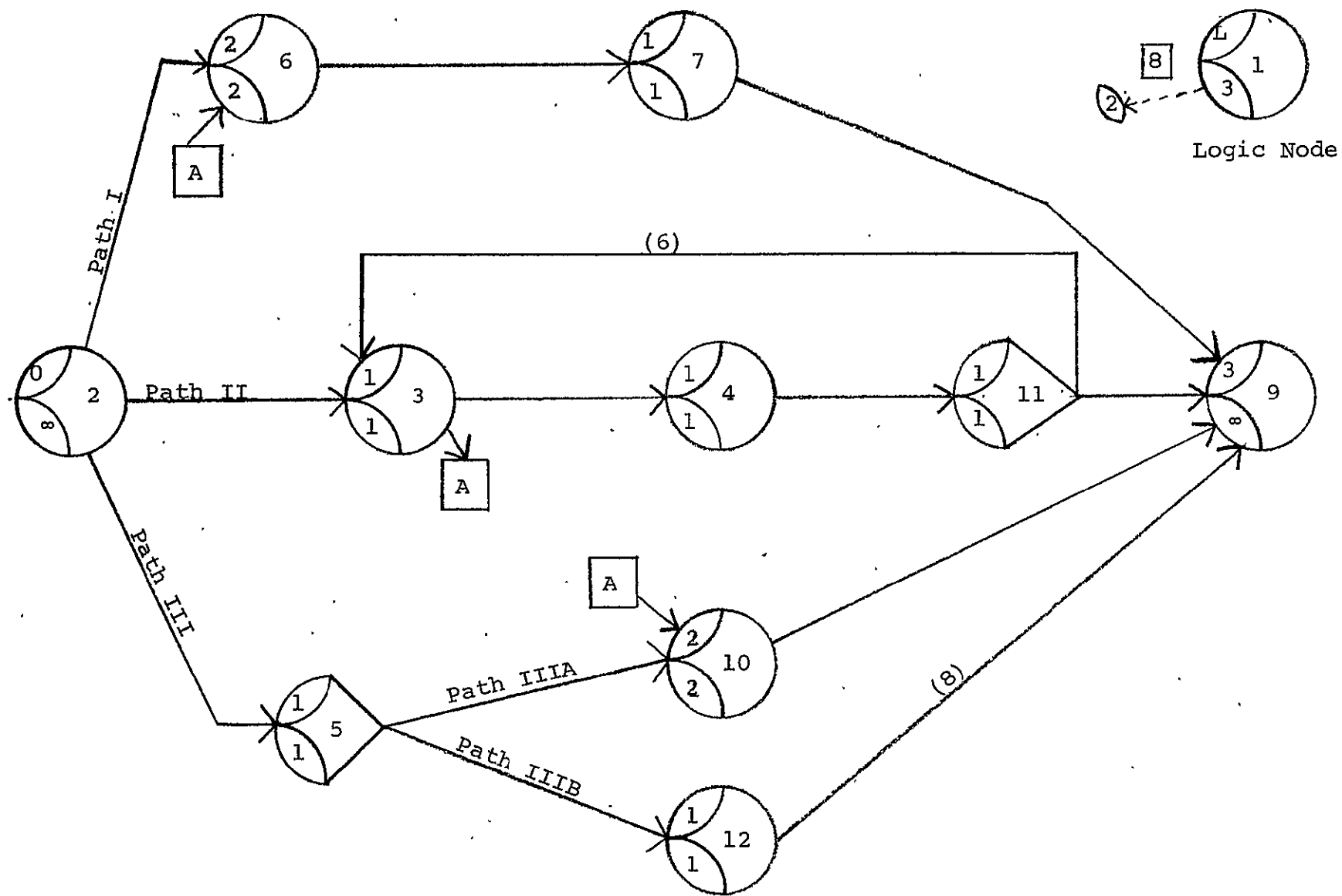


Figure 5. Example of the Use of the Logic Node Reset.

The cost of each activity is assumed to be a linear function of time,

$$\text{Cost} = a + b (t_1 - t_0),$$

where $(t_1 - t_0)$ is the elapsed activity time, b is a parameter which accounts for costs which vary linearly with elapsed time, and a is a constant or set-up cost.

Costs are accumulated for each activity performed up to the time of realization of a project end node. In cases where activities in progress are terminated, costs up to the time of termination are included in the total project cost.

3.3 Variance Reduction Techniques

Even with third generation computers, the cost of obtaining parameter estimates using the GERT-II Simulator is not insignificant. For this reason, it is natural to consider the use of Monte Carlo variance reduction techniques. One of the most promising approaches in this case is the use of an antithetic variable. A description and evaluation of this approach is given in Appendix II. The results of this study indicate that this approach is effective on PERT type networks, consisting entirely of and node input and deterministic output. In this case, the estimate of the mean duration of the project is improved but only at the expense of a less efficient estimate of the variance of project duration times. The results obtained on GERT networks with exclusive or node inputs and probabilistic outputs were not encouraging, and the use of the antithetic variable is not recommended in these cases.

4. Application of GERT to Development-Type Projects

The effect of adding project activity data such as time, cost, and branching probabilities, will be treated in this paper by considering a development-type project, characteristic of industries that regularly turn out new models of a product. Although our example is taken from the weapons industry, the concepts and techniques used in its analysis can be directly applied to the management of other development projects such as those found in the automotive and electronic industries.

Our illustrative example consists of three subassemblies to be developed concurrently. These subassemblies are of a general nature. For example, the chassis, the power train, and the body might be the subassembly breakdown used in the automotive industry. In our example, they are the submunition, the dispenser, and the fuse.

At several stages of product development, management must assure that the subassemblies are compatible. Precedence relationships are thus built in the network model to signify that development of a particular subassembly must be preceded at various stages by the design or partial design of other subassemblies. Finally, testing of prototypes may result in partial or total redesign of a subassembly which in turn may result in returning to an earlier development stage on other subassemblies.

Weapons System Development Project. To illustrate the above concepts, the development of Air Force munitions systems

is modelled using GERT-IIA nomenclature. The complete network, which contains about 150 nodes and 200 activities, is sufficiently complex to require simulation for proper evaluation. A simplified version of the network is shown in Figure 6, with details of the three subassemblies being given in Figures 6A, 6B and 6C. Each of the three subassemblies provide for the choice of one of three paths depending on the decision to, (1) develop a completely new design, (2) modify an existing unit, or (3) use an existing unit. Progress in each of the three subassemblies is interrelated by "approval" signals. For example, if node 59 in Figure 6B is realized, approval signal D_2S is initiated. (The nomenclature adopted in this network is as follows. The symbol D_2S denotes that the second signal from the Dispenser subassembly goes to the Submunition subassembly.) The approval signal D_2S then enters one of the nodes 101, 87, 85 or 78 in the submunition subassembly, Figure 6A. It is sent to the proper node by using the node switching routine previously illustrated in Figure 3. In this case, the choice of paths out of node 76 (and possibly 83 as well) controls the node changing routine so that the approval signal D_2S is sent to the proper node. Special nodes to control the flow of these signals are not shown in Figure 6.

The standard management policy used in the development of new munition systems is depicted by the flow of approval signals shown in Figures 6, 6A, 6B and 6C. This will be denoted as Policy 1 below. A partial listing of the computer output for this policy is shown in Figure 7.

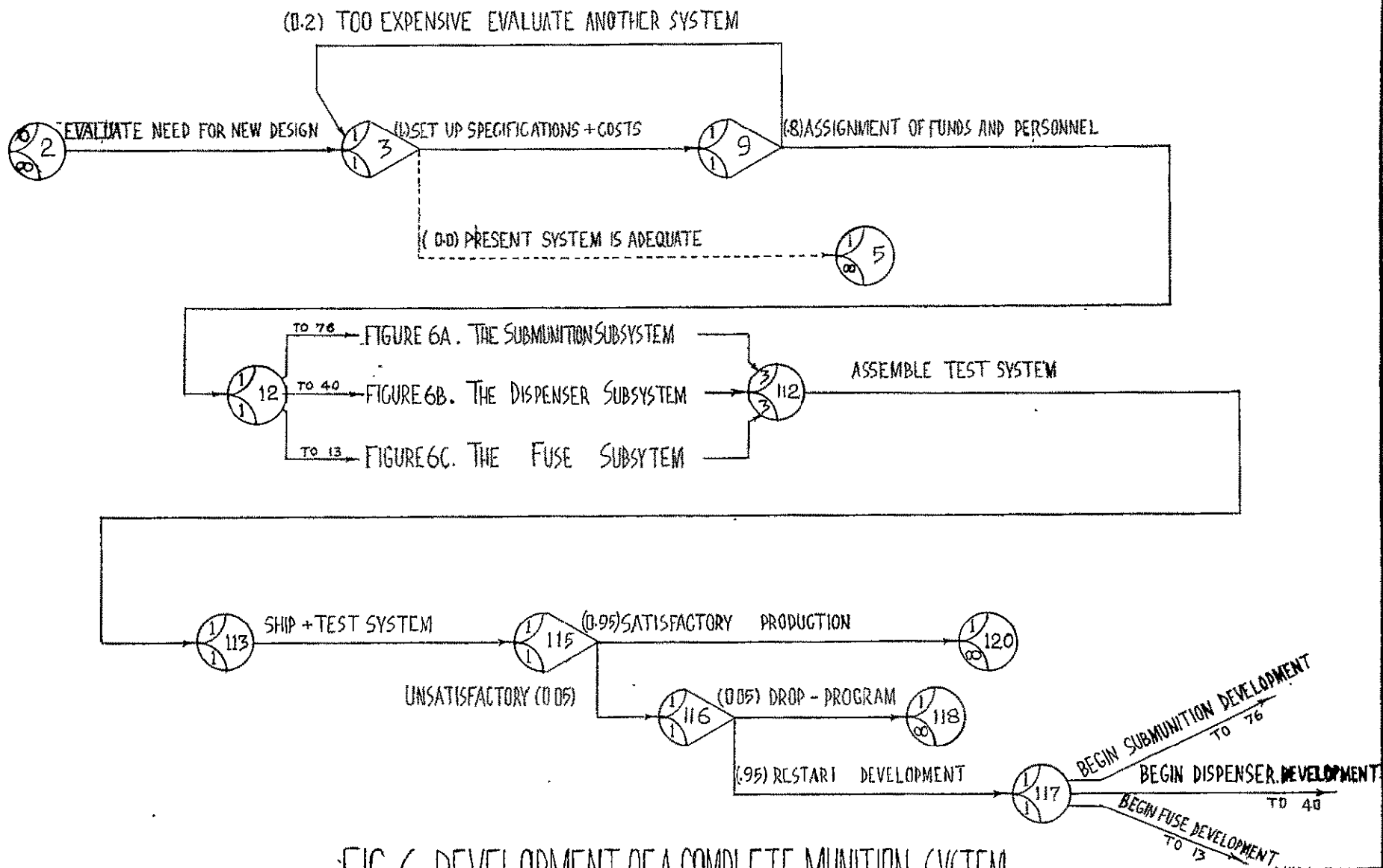


FIG.6: DEVELOPMENT OF A COMPLETE MUNITION SYSTEM

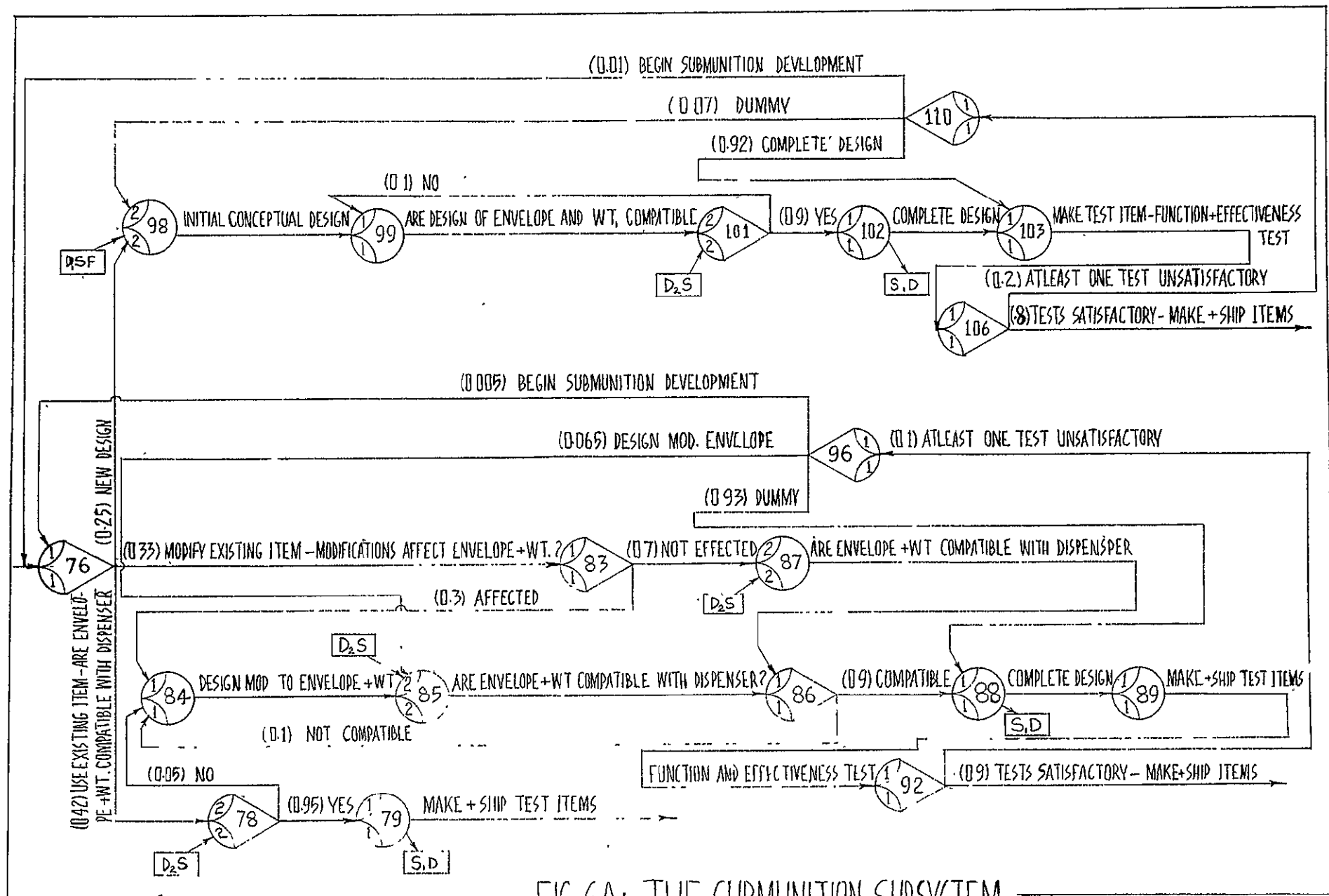


FIG.6A: THE SUBMUNITION SUBSYSTEM



FIG. 6C: THE FUSE SUBSYSTEM

GERT SIMULATION PROJECT 116 BY CLARK
DATE 4/ 23/ 1970

FINAL RESULTS FOR 200 SIMULATIONS

NODE	PROB./COUNT	MEAN	STD.DEV.	MIN.	MAX.	STD.ER. OF MEAN
5	0	NO VALUES RECORDED				
120	1.0000	103.2355	49.7300	54.3450	363.2366	3.5164
118	0	NO VALUES RECORDED				

HISTOGRAMS

NODE	LOWER LIMIT	CELL WIDTH	FREQUENCIES											
5	0.0	0.30	0	0	0	0	0	0	0	0	0	0	0	0
120	20.00	10.00	0	0	0	0	7	23	31	34	28	30	14	
			6	7	4	2	1	1	0	2	2	0	1	
			0	0	1	0	0	0	0	3	1	2		
118	20.00	4.00	0	0	0	0	0	0	0	0	0	0	0	
			0	0	0	0	0	0	0	0	0	0	0	
			0	0	0	0	0	0	0	0	0	0	0	

HISTOGRAMS

NODE	LOWER LIMIT	CELL WIDTH	FREQUENCIES											
COST	0.0	50.00	0	0	0	0	0	16	23	21	30	27	10	
			16	18	8	5	3	0	0	3	0	0	0	
			2	1	1	0	0	0	1	0	0	6		

AVG. COST PER REALIZATION 496.02
STANDARD DEVIATION 336.23
MIN. COST FOR REALIZATION 204.99
MAX. COST FOR REALIZATION 2326.43

NODE	START	END	WITH ATTRIBUTES				TIME/END NODE	TIME/ACTIVITY	PERCENT	
2	3	3	4	0	1	10	1	3.46	3.46	1.00
3	6	2	4	0	2	12	8	5.51	2.05	1.00
3	4	33	1	0	3	0	0	0.0	0.0	1.00
4	5	33	1	0	4	0	0	0.0	0.0	0.0
6	7	20	4	0	5	1	1	6.45	0.94	1.00
7	8	18	4	0	6	3	8	12.23	5.78	1.00
9	9	2	4	0	7	0	1	14.47	2.24	1.00
9	11	21	4	0	10	0	1	21.20	6.73	1.00
9	10	2	4	0	8	0	1	16.10	1.63	0.21
10	3	3	4	0	9	10	1	3.46	-12.64	1.00
11	12	24	4	0	11	1	1	26.34	5.13	1.00
12	13	1	4	0	12	2	3	27.41	1.08	1.00
12	40	1	4	0	193	3	6	27.37	1.03	1.00
12	76	1	4	0	99	3	3	27.43	1.09	1.00
13	14	33	1	0	29	0	4	29.97	2.55	0.44
13	27	5	4	0	35	3	8	32.86	5.45	0.35
13	17	2	4	0	13	2	4	33.64	6.23	0.25
14	15	2	4	0	32	0	1	32.12	2.16	0.44
14	16	2	4	0	33	0	1	32.17	2.20	0.44
14	121	33	1	0	0	0	0	29.18	-0.78	0.18
15	125	33	1	0	0	0	0	36.86	4.74	0.02
16	112	33	1	0	103	0	0	82.05	49.83	1.00
17	18	1	4	0	14	3	4	35.98	2.34	0.25
18	19	3	4	0	15	3	3	39.37	3.38	0.25
18	20	4	4	0	16	1	1	36.64	0.66	0.25
19	121	33	1	0	0	0	0	29.18	-6.80	0.18
19	21	3	4	0	17	2	3	42.38	3.02	0.25
20	24	2	4	0	21	0	1	38.71	2.06	0.25
21	22	5	4	0	18	2	2	45.45	3.06	0.25
21	26	2	4	0	19	0	1	46.36	3.93	0.25
22	26	33	1	0	22	0	0	46.36	0.91	0.25
22	23	33	1	0	23	0	0	44.38	-1.07	0.02
23	19	3	4	0	27	1	0	39.37	-5.01	0.25
23	18	1	4	0	28	3	0	35.98	-8.39	0.25
23	13	1	4	0	34	2	2	27.41	-16.97	1.00
24	125	33	1	0	0	0	0	36.86	-1.85	0.02
24	112	33	1	0	102	0	0	82.05	35.69	1.00
27	28	7	4	0	36	2	6	48.15	15.28	0.35
28	29	3	4	0	37	2	5	51.62	3.48	0.35
29	30	24	4	0	38	1	3	56.91	5.29	0.35
29	31	6	4	0	39	0	1	52.42	0.80	0.35
29	121	33	1	0	0	0	0	29.18	-22.44	0.18
30	33	3	4	0	41	1	1	60.58	3.68	0.35
31	32	2	4	0	40	0	1	54.63	2.21	0.35
32	125	33	1	0	0	0	0	36.86	-17.76	0.02
33	36	2	4	0	43	0	1	67.86	7.28	0.35
33	34	5	4	0	42	2	2	63.83	3.25	0.35
34	36	33	1	0	45	0	0	67.86	4.03	0.35
34	37	33	1	0	49	0	0	67.71	3.88	0.06
36	112	33	1	0	102	0	0	82.05	14.19	1.00
37	30	9	4	0	53	1	2	56.91	-10.81	0.35
37	28	7	4	0	51	2	4	48.15	-19.57	0.35
37	29	8	4	0	52	2	3	51.62	-16.09	0.35
37	13	1	4	0	50	2	2	27.41	-40.30	1.00
40	45	33	1	0	59	0	0	28.82	1.45	0.51

Of particular interest in making a preliminary evaluation of alternative management policies is the listing of the mean time to the first realization of each node. For example, it might be proposed that the approval signals emanating from nodes 28 (F_1D) and 32 (F_2D) in the fuse subassembly be combined and issued from node 36 (Policy 2 below) as signal F_1D . The purpose of this change would be to hold up the dispenser development until the fuse passes final inspection, since the latter has a relatively high probability of failure. This change would then avoid having to repeat the dispenser work when an unsatisfactory test occurs in the fuse subassembly.

From the output of the type given at the bottom of Figure 7, the following average first node realization times could be determined.

<u>Node No.</u>	<u>Ave. Realization Time</u>	<u>Node No.</u>	<u>Ave. Realization Time</u>
28	48.2	42	37.3
32	54.6	46	38.3
36	67.9	59	56.3

From these times, it can be concluded that this policy change will cause a significant increase in the total project duration, since the time for node 36 is considerably later than node 28 which was already later than nodes 42 and 46. The node branching probabilities should, of course, also be considered in making evaluations of this type.

Two potentially useful modifications of the standard management policy are described below. Policies 2 and 3 are more conservative than Policy 1; they should result in less

duplication of work, and thus less cost. This cost reduction, of course, is at the expense of increased project duration time which results from "holding up" the issuance of approval signals, or requiring special approvals not used in the standard policy, both of which potentially avoid duplication of work.

Policy 1: Standard development program policy as shown in Figures 6, 6A, 6B and 6C.

Policy 2: Modification of Policy 1 in which the approval signals to develop the dispenser assembly (F_1D and F_2D) are delayed and combined, and now emanate from node 36 instead of nodes 28 and 32, as signal F_1D .

Policy 3: Modification of Policy 1 in two parts. First, it delays development of the submunition assembly by moving D_2S from node 101 to 99 and from nodes 85 and 87 to 83. This also eliminates the need for the corresponding feedback loops. Also, it holds up the fabrication of the submunition assembly and the testing and shipping of the actuator until the dispenser development is completed. The latter is accomplished by holding up the realization of actuator nodes 33, 19 and submunition nodes 89, 103 until dispenser nodes 50 or 71 are realized.

Table 1. Results of Simulation of Three Policies for the Development of a Munitions System, As Shown in Figure 6.

Mean Values of the Runs

Policies	Times			Costs		
	(1)	(2)	(3)	(1)	(2)	(3)
Run 1	99.2	108.7	109.7	496.0	478.6	487.2
Run 2	106.6	110.2	109.9	468.7	503.3	479.6
Run 3	99.5	114.3	107.5	515.5	516.4	486.0
Run 4	104.8	102.9	106.9	466.1	453.1	467.6
Run 5	103.2	103.4	107.4	514.4	454.2	461.7
Run 6	103.2	109.5	104.7	496.0	470.1	456.0
Run 7	99.8	110.9	108.1	474.6	492.8	472.3
Run 8	102.9	114.8	115.5	498.1	508.9	519.3
Run 9	99.1	107.2	108.9	467.5	484.3	474.9
Run 10	104.2	112.0	98.8	498.4	497.4	429.5
Average*	102.3	109.4	107.7	489.5	485.9	473.4
Variance*	7.1	16.3	17.8	357.8	480.9	544.3
L.S.D.** ($\alpha=.05$)	3.40			19.7		
($\alpha=.10$)	2.82			16.4		

Statistics on Single Network Realizations

Standard Deviations Among the 200 Simulations Within a Run

Policies	Times			Costs		
	(1)	(2)	(3)	(1)	(2)	(3)
Run 1	37.2	46.6	38.2	336.2	253.7	238.0
Run 2	46.1	52.5	43.5	239.2	295.0	242.3
Run 3	36.9	62.3	41.9	303.6	369.2	270.2
Run 4	48.9	36.2	41.0	209.9	182.7	244.7
Run 5	49.7	45.8	35.1	330.3	235.5	203.2
Run 6	49.7	44.1	39.7	336.2	215.0	203.3
Run 7	47.3	66.6	51.4	303.0	351.3	326.5
Run 8	48.7	56.1	46.7	311.1	329.7	301.7
Run 9	33.9	48.4	49.0	203.4	265.1	290.9
Run 10	42.9	55.8	29.4	279.7	312.8	168.4
Average*	44.1	51.4	41.6	285.3	281.0	248.9
Variance*	36.2	82.3	43.0	2563.3	3709.7	2428.4
L.S.D.* ($\alpha=.05$)	6.20			49.4		
($\alpha=.10$)	5.15			40.9		
(Run) Min.	51	50	51	198	196	196
(Averages) Max.	354	470	377	2230	2490	2230
(All)	46	44	47	186	184	184
(Runs)	456	814	572	2830	4300	3830

*The averages and the variances were computed from the corresponding 10 run values.

**L.S.D.=Least Significant Difference= $t_{\alpha/2} \sqrt{2(\text{Error Var.})/\text{No.Obs. in Ave.}}$
 For the Mean Time, Error Var.=(7.1+16.3+17.8)/3=13.7, and for $\alpha=.05$
 L.S.D.=2.052 $\sqrt{2 \times 13.7/10}$ =3.40.

Each of the above policies was evaluated using the GERT-IIA Simulator program. The data shown in Table 1 give the results of 30 computer runs, 10 for each of the 3 policies. Each run in turn consisted of 200 random realizations of the project network. The durations of the project activities were assumed to be constants or to have a Gamma distribution. The form of the input data, along with an example, is given in Appendix III.

The top half of Table 1 summarizes the mean values for time and cost for each of the 30 runs. The grand averages of the 10 runs for each policy indicate, as expected, that policies 2 and 3 require more time on the average than policy 1, but at reduced cost. Statistical tests using both the Duncan Multiple Range test and the Least Significant Difference method were performed. (See Reference 4) In both cases, no differences were found between the mean times for policies 2 and 3, but they both differ from policy 1, at the 5 percent level. With regards to mean costs, policy 3 is significantly lower than 1 at about the 10 percent level.

The lower half of Table 1 summarizes the standard deviations of time and cost, among the 200 network simulations which constitute each run. Policy 3 has the lowest standard deviation for both time and cost. While the differences are not statistically significant in all cases there is reason to believe that Policy 3, which reduces duplication of work, would have less variability among total project duration times.

Table 1 also gives the range of project duration times. The first set of values are the averages of the ranges for the

10 runs, while the second set is the range of values over all 10 runs.

The following conclusions could be drawn from this computer simulation study.

- (1) Since Policy 3 has a lower cost, a lower variability among project duration times, and probably a lower mean project duration than Policy 2, the latter is not a desirable modification of Policy 1.
- (2) Compared with Policy 1, Policy 3 reduces the expected project cost by about 3 percent with a corresponding increase in project duration of 5 percent. This may or may not be a desirable change.
- (3) The observed range of project duration times, as well as the standard deviations given in Table 1, indicate that a great deal of uncertainty exists in the total project duration time and cost. It undoubtedly would be desirable to find ways to reduce this uncertainty.
- (4) The computer (CPU) time required for 10 runs as shown in Table 1 was approximately 6 minutes on an IBM 360/65 system, with a corresponding total cost of about \$30. This is considered to be reasonable, and is considerably less than the cost of using other simulation languages such as GPSS.
- (5) Summary

The use of GERT network nomenclature to describe a project plan or system operating policy was considered. The advantages of using this approach in system analysis,

design, and ultimately in communication were described. The flexibility of the GERT nomenclature, compared with the quite restrictive logic of PERT/CPM, makes this a tool of considerable utility. Finally, if estimates of time, cost, branching probabilities, etc. can be obtained to describe the network activities, the GERT simulation program provides a feasible way to both propose and evaluate system changes.

REFERENCES

1. Abernathy, W. J. and R. S. Rosenbloom, "Parallel and Sequential R and D Strategies: Application of a Simple Model", IEEE Transactions on Engineering Management, EM-15, No. 1, 2-10, March 1968.
2. Pritsker, A. B. and Ishmael, P. C. Gert Simulation Program II, June 1969, NASA Electronics Research Center.
3. Tocher, K. D. The Art of Simulation. Longon: The English Universities Press, 1963.
4. Duncan, A. J., "Quality Control and Industrial Statistics," R. D. Irwin, Inc. 1959, pp. 601-4.

APPENDIX I

	<u>Pages</u>
1. Purpose of Each Subroutine	I-1
2. Calling Sequence	I-3
3. Program	I-4

I-1

Listing of Subroutines and the Purpose of Each

1. Main Initializes reader, printer, and other variables.
2. Aloop Cancels activities in process from the nodes specified and adds the partial cost of any activities removed.
3. Antim Stores antithetic times and antithetic branching probabilities.
4. Colct Stores statistics on the time that a node has been realized for the sink nodes and any others specified as input.
5. Datan Reads in all data from the data cards and outputs what was read in.
6. Drand Random number generator.. If seed read from the data card is zero, it will automatically start with 1267.
7. Error Prints out error messages. If called from Datan stops because all data has not been read in, and there is no way, as yet, to tell where one data set ends and another begins. If all data has been read in, it will restart the program with the next data set.
8. Filem Used to file all activities in the activity file and to set up the predecessor and successor pointers.
9. Gasp The most important routine which calls all other routines directly or indirectly (except main). This routine is the "supervisor" of the entire program.

10. Histo Stores histogram information for time and cost.
11. Nfind Used to find an activity entry in the activity file and to return the location of the activity.
12. Rmove Removes an activity from the file and resets the pointers.
13. Rnorm Routine for generating normal times.
14. Sampl Routine for generating the time distributions allowed in the program both for antithetic and non-antithetic times.
15. Schat Schedules activities from a node which has just been realized or the source nodes, and puts the activities in the activity file. Coordinates antithetic and non-antithetic branching.
16. Sumry Calculates and summarizes the results and outputs them.

SUBROUTINE CALLING CONVENTIONS

ROUTINE	CALLS	CALLED BY
MAIN	GASP	
ALoop	NFind, RMove	GASP
ANTIM		SAMPL, SCHAT
COLCT	ERROR	GASP
DATAN	DRAND, ERROR, FILEM	GASP
DRAND		DATAN, SAMPL, SCHAT
ERROR	GASP	COLCT, DATAN, FILEM, GASP HISTO, NFind, RMove, SCHAT, SUMRY
FILEM	ERROR	DATAN, GASP, SCHAT
GASP	ALoop, COLCT, DATAN, ERROR, FILEM, HISTO, NFind, RMove, SCHAT, SUMRY	MAIN, ERROR
HISTO	ERROR	GASP
NFind	ERROR	ALoop, GASP
RMove	ERROR	ALoop, GASP
RNORM		SAMPL
SAMPL	ANTIM, DRAND, RNORM	SCHAT
SCHAT	ANTIM, DRAND, ERROR, FILEM, SAMPL	GASP
SUMRY	ERROR	GASP

FORTRAN IV G LEVEL 1, MOD 4

MAIN

DATE = 70180

13/46/04

PAGE 0001

```

0001      DIMENSION NSET(5000),QSET(500)
0002      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0003      COMMON MXC,ATRI8,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0004      COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0005      COMMON KOL,KOF,KLE
0006      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0007      COMMON NTYP(500),MX,NFTBU(500),CSTAR,CINC
0008      COMMON NNM,IMN,NPD,NQ,NPRMS,NPUNS,NYR,MAXNS,MAXA,NPRTOP,
0009      INAME(6),NPROJ,MON,NDAY,NSKST,NSRC,NREL(500),NREL(500),NSKSR,
0010      2NSORC(20),TOTIM,KOUNT(4),JSINK(20),NPD(3,500),NABA(3,800)
0011
0012      C*****SET EQUIPMENT NUMBERS FOR CARD READER(NCRDR) AND PRINTER (NPRNT). GRTS 90
0013      C*****SET NUMBER OF ATTRIBUTES AND FILE INDICATORS FOR NSET
0014      C
0015      NCRDR=5
0016      NPRNT=6
0017      IM=8
0018      KOL = 7777
0019      KOF = 8888
0020      KLE = 9999
0021      IMN=800
0022
0023      C*****ANTITHETIC STORAGE
0024      C
0025      7      IMM=4
0026      CALL GASP (NSET,QSET)
0027      GO TO 7
0028      END

```

GRTS 100
GRTS 110
RMVS 110
PMVS 120
RMVS 130
GRTS 140
GRTS 150

NOT REPRODUCIBLE

FORTRAN IV G LEVEL 1, MOD 4

ALLOOP

DATE = 70180

13/46/04

PAGE 0001

```

0001      SUBROUTINE ALLOOP (NSET,QSET,NPOS,COST,ACTIME,JJJ)
0002      DIMENSION NSET(1),QSET(1),ACTIME(1)
0003      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRI8,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008      COMMON NTYP(500),MX,NFTBU(500),CSTAR,CINC
0009      COMMON NNM,IMN,NPD,NQ,NPRMS,NPUNS,NYR,MAXNS,MAXA,NPRTOP,
0010      INAME(6),NPROJ,MON,NDAY,NSKST,NSRC,NREL(500),NREL(500),NSKSR,
0011      2NSORC(20),TOTIM,KOUNT(4),JSINK(20),NPD(3,500),NABA(3,800)
0012      KNODE=NPOS
0013      1      INODE=NABA(3,NPOS)
0014      IF(INODE.LE.0) GO TO 11
0015      2      KCOL=NFIN(INODE,1,1,NSET,QSET)
0016      IF(KCOL.LE.0) GO TO 5
0017      CALL RMVFKCOL(1,NSET,QSET,NAT,KOWT)
0018      JA=JTRIB(1)
0019      TIME=TNOW-ACTIME(JA)
0020      COST=COST+FLOAT(JTRIB(8))+FLOAT(JTRIB(7))*TIME
0021      GO TO 2
0022      5      NPOS=NPOS+1
0023      GO TO 1
0024      11      IF(NPROJ.GE.0) RETURN
0025      IF(NRUN.LT.NABA(1,IMN).OR.NRUN.GT.NABA(1,IMN-1)) RETURN
0026      10      NPOS=NPOS+1
0027      WRITE(NPRNT,10) (NABA(3,K),K=KNODE,NPOS)
0028      FORMAT(15X,' ACTIVITIES EMANATING FROM THE FOLLOWING NCDES HAVE SE-
0029      LUF DESTROYED',/(24(5))
0030      RETURN
0031      END

```

FORTRAN IV G LEVEL 1, MOD 4

ANTIM

DATE = 70180

13/46/04

PAGE 0001

```

0001      SUBROUTINE ANTIM(INO,NEXT,STORE,TIBR)
0002      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0003      DIMENSION STORE(1)
0004      C
0005      C***** IND=1 GENRATE AND STORE
0006      C***** 2 USE AND ZERO OUT
0007      C
0008      INDX=(NEXT-1)*IMM+1
0009      GO TO (1,2),IND
0010      1      DO 5 I=1,IMM
0011      IF(STORE(INDX).EQ.0.0) GO TO 6
0012      INDX=INDX+1
0013      WRITE(NPRNT,15) NEXT,IMM
0014      15      FORMAT(2I10,'OVERFLOW ANTITHETIC')
0015      RETURN
0016      STORE(INDX)=TIBR
0017      RETURN
0018      DO 10 I=1,IMM
0019      IF(STORE(INDX).NE.0.0) GO TO 11
0020      INDX=INDX+1
0021      GO TO 12
0022      11      TIBR=STORE(INDX)
0023      STORE(INDX)=0.0
0024      RETURN
0025      12      TIBR=0.0
0026      RETURN
0027      END

```

FORTRAN IV S LEVEL 1, MOD 4 COLCT DATE = 70180 13/46/04 PAGE 0001

```

0001 SUBROUTINE COLCT (X,N)
0002 COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0003 COMMON MXC,ATRI8 ,JCELS(20,32),SUMA(100,5),NHIST,NCRDP
0004 COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0005 IF(N.GT.0.AND.N.LE.NCLCT) GOTO 7
0006 CALL ERROR(4,NSET,QSET)
0007 3 SUMA(N,1) = SUMA(N,1)+X CLTS 100
0008 SUMA(N,2) = SUMA(N,2)+X*X CLTS 110
0009 SUMA(N,3) = SUMA(N,3)+1.0 CLTS 120
0010 SUMA(N,4) = AMINI(SUMA(N,4),X)
0011 SUMA(N,5) = AMAXI(SUMA(N,5),X)
0012 RETURN CLTS 150
0013 END CLTS 160

```

FORTRAN IV G LEVEL 1, MOD 4 DATAN DATE = 70180 13/46/04 PAGE 0001

```

0001 SUBROUTINE DATAN (NSET,QSET,ATIME,BRNCH,ACTIME,LUPDO)
0002 DIMENSION NSET(1),QSET(1),ATIME(1),BRNCH(1),ACTIME(1)
0003 COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004 COMMON MXC,ATRI8 ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005 COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0006 COMMON KOL,KOF,KLE
0007 COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008 COMMON NTYPE(500),MX,NFTBU(500),CSTAR,CINC
0009 COMMON NNM,IMN,NPD,NQ,NPRMS,NRUNS,NYR,MAXNS,MAXA,NPRTOP ,
INAME(6),NPROJ,MON,NDAY,NSKST,NSRC,NRELP(500),NREL(500), NSKSR,
2NSORC(20),TOTIM,KOUNT(4),JSINK(20),NPO(3,500),NABA(3,800)
0010 NRUN = 1 DATS 100
0011 READ(NCRDR,101,END=9999)NAME,NPROJ,MON,NDAY,NYR,NRUNS,NPRMS,ID,
1JSEED,CSTAR,CINC
0012 101 FORMAT(6A2,I4,2I2,4I4,I10,2F10.0)
0013 IF(NPRMS.GT.300) CALL ERROR(1,NSET,QSET)
0014 IF(NRUNS) 30,30,5 DATS 130
0015 30 WRITE(NPRNT,241)
0016 241 FORMAT(/' NRUNS = 0 ***ON INPUT')
0017 CALL ERROR (1,NSET,QSET)
0018 10 FORMAT(25I3) DATS 150
0019 14 FORMAT(8F10.2) DATS 160
C
C*****READ LARGEST NODE NUMBER, NO. OF SINK NODES, NO. OF SOURCE NODES, DATS 170
C*****NO. OF NODES NECESSARY TO REALIZE THE NETWORK,NODES MOD.,PRT CODE DATS 180
C
0020 5 READ(NCRDR,10)NOQ,NSRC,NPD,NSKST,NSKS,NCTS,NNM,NNR,LGOP,ITIME,
1IBRNCH,LUPDO,NPRTOP,LOGIC
0021 IF(NSRC.GT.20) CALL ERROR(1,NSET,QSET)
0022 IF(NSKS.GT.20) CALL ERROR(1,NSET,QSET)
0023 NSKSR=NSKST DATS 200
C
C*****READ SOURCE AND SINK NODE NOS. DATS 210
C
0024 READ(NCRDR,10) (NSORC(J),J=1,NSRC), (NSINK(K),K=1,NSKS) DATS 220
0025 IF(LOGIC.EQ.0) GOTO 13
0026 READ(NCRDR,10) NRELP(1),NTYPE(1)
0027 GOTO 16
0028 13 NRELP(1)=0
0029 NTYPE(1)=0
C
C*****READ NO. OF RELEASES FOR EACH NODE. NODE NOS. MUST BE .GE. TO 2. DATS 230
C
0030 16 READ(NCRDR,10) (NRELP(I),I=2,NOQ)
0031 DO 11 I=2,NOQ
0032 11 NREL(I)=NRELP(I) DATS 260
C
C*****READ NODE TYPE. 1 AND 3 ARE DETERMINISTIC. 2 AND 4 ARE PROBABILIDATS 270

```

FORTTRAN IV G LEVEL 1, MOD 4 DATAN DATE = 70180 13/46/04 PAGE 0002

```

C*****3 AND 4 CAN BE REALIZED MORE THAN ONCE.      DATS 280
C
0033      READ(NCRDR,10)(NTYPE(I),I=2,NOQ)
C
C*****READ LOWER LIMIT AND CELL WIDTH FOR HISTOGRAMS.      DATS 300
C
0034      READ(NCRDR,14)(XLOW(K),K=1,NSKS)      DATS 310
0035      READ (NCRDR,14)(WIDTH(K),K=1,NSKS)      DATS 320
0036      IF(NCTS) 135,135,125      DATS 330
0037      125 DO 130 J=1,NCTS      DATS 340
0038      130 KOUNT(J)=0      DATS 350
0039      135 DO 140 K=1,NSKS      DATS 360
0040      140 JSINK(K)=1      DATS 370
0041      NHIST=NSKS      DATS 380
0042      NCLCT=NSKS*(NCTS+1)      DATS 390
0043      MXC=32      DATS 420
0044      DO 9 I = 1,NPRMS      DATS 450
0045      9 READ (NCRDR,106) (PARAM(I,J),J=1,4)      DATS 460
0046      106 FORMAT(4F10.4)      DATS 470
0047      ISEED=JSEED      DATS 490
0048      RNUM=DRAND(ISEED)      DATS 500
0049      TNDW=0.0      DATS 510
C
C*****INITIALIZE NSET      DATS 520
C
0050      MX = IM +1      DATS 560
0051      MXX=IM+2      DATS 570
0052      MAXNS = ID * MXX      DATS 590
0053      MAXA=ID*IMM
0054      DO 35 J=1,ID
0055      ACTIME(J)=0.0
0056      35 QSET(J)=0.0      DATS 610
0057      DO 40 J=1,MAXNS      DATS 620
0058      40 NSET(J)=0      DATS 630
0059      INDX=0
0060      DO 45 I=1,ID      DATS 640
0061      INDX=INDX+MXX
0062      NSFT(INDX - 1) = I + 1      DATS 660
0063      45 NSET(INDX)=I-1      DATS 670
0064      NSET(MAXNS - 1) = KOF      DATS 680
0065      DO 3 K = 1,NOQ      DATS 690
0066      NQ(K)=0      DATS 700
0067      MFE(K)=0      DATS 710
0068      3 MLF(K)=0      DATS 720
0069      MFA=1
0070      ATRIB=0.0
0071      DO 301 K=1,IM
0072      301 JTRIB(K)=0

```

FORTTRAN IV G LEVEL 1, MOD 4 DATAN DATE = 70180 13/46/04 PAGE 0003

```

0073      CALL FILEM(1,NSET,QSET)      DATS 790
0074      DO 320 JS = 1,10
C
C*****DATA CARD TYPES NINE      DATS 810
C
0075      READ(NCRDR,1110) ATRIB , (JTRIB(JK),JK=1,IM)
0076      JQ=JTRIB(1)
0077      1110 FORMAT(F8.3,9I3)
0078      IF(JQ)23,15,320      DATS 840
0079      320 CALL FILEM(JQ,NSET,QSET)      DATS 860
0080      15 IF(NCLCT.LT.0.OR.NHIST.LT.0) GOTO 23
0081      20 WRITE(NPRNT,21)      DATS 980
0082      WRITE(NPRNT,22) NPROJ,NAME,MON,NDAY,NYR      DATS 990
0083      WRITE(NPRNT,24)      DATS1000
0084      DO 80 JQ=2,NOQ      DATS1010
0085      PROB = 0.0      DATS1020
0086      LINE = MFE(JQ)      DATS1030
0087      NT = NTYPE(JQ)      DATS1040
0088      IF(LINE)80,80,82      DATS1050
0089      82 IB = (LINE-1) * MXX + 1      DATS1060
0090      IE = IB + IM-1      DATS1070
0091      DPROB=QSET(LINE)
0092      GO TO (72,73,72,73,72,73,72,73),NT      DATS1100
0093      73 QSET(LINE)=QSET(LINE)+PROB
0094      PROB = PROB + DPROB      DATS1120
0095      72 WRITE(NPRNT,12) (NSET(I),I=IB,IE),DPROB
0096      LINE = NSET(IE+1)      DATS1140
0097      IF(LINE - 7777) 82,80,84      DATS1150
0098      84 CALL ERROR(2,NSET,QSET)
0099      80 CONTINUE      DATS1170
0100      WRITE(NPRNT,26)      DATS1180
0101      DO 50 I = 1,NPRMS      DATS1190
0102      50 WRITE(NPRNT,28) I,(PARAM(I,J),J=1,4)      DATS1200
0103      21 FORMAT(1H1)      DATS1210
0104      22 FORMAT(32X,23HGERT SIMULATION PROJECT,I4,4H BY ,6A2/45X,4HDATE,      DATS1220
         1I3,1H/,13,1H/,15//)      DATS1230
0105      24 FORMAT(42X,23H**NETWORK DESCRIPTION**//16X5HSTART,6X3HEND,6X,      DATS1240
         19HPARAMETER,6X,12HDISTRIBUTION,6X,5HCOUNT,6X,8HACTIVITY,6X,      DATS1250
         A'VAR. COST',4X,'FIX. COST',3X,
         2 11HPROBABILITY/
         317X4HNODE,6X,4HNODE,7X,6HNUMBER,11X4HTYPE,11X4HTYPE,7X,6HNUMBER/)      DATS1260
0106      12 FORMAT(119,11I,112,116,414,F17.4)      DATS1270
0107      199 FOPMAT(//36X26HERROR EXIT, TYPE 70 ERROR.)      DATS1290
0108      26 FORMAT(//39X23H**ACTIVITY PARAMETERS**//16X9HPARAMETER32X,10HPARAM      DATS1300
         1ETERS/17X6HNUMBER,16X1H1,13X1H2,13X,1H3,13X1H4/)      DATS1310
0109      28 FORMAT(121,7X,4F14.4)      DATS1320
0110      DO 1024 I=1,3
0111      DO 1024 J=1,10

```

FORTRAN IV G LEVEL 1, MOD 4 DATAN DATE = 70180 13/46/04 PAGE 0004

```

0112      1024  NPD(I,J)=0
0113          NP=1
0114          IF(NNM.LE.0) GOTO 1020
0115      1025  WRITE(NPRNT,1004)
0116      1004  FORMAT(/39X25H**NETWORK MODIFICATIONS**// 2X,8HACTIVITY,12(1X,9HNDATSI390
          100E FILE)/)
          DATSI400
C
C*****NETWORK MODIFICATIONS STORED IN NABA(1,ACTIVITY NUMBER)
C*****NODE RESETTING STORED IN NABA(2,ACTIVITY NUMBER)
C*****LOOPING STORED IN NABA(3,ACTIVITY NUMBER)
C
0117          NPOS=1
0118      1026  IE=NPOS+23
0119          IF(IE.GT.IMN) CALL ERROR(1,NSET,QSET)
0120          READ(NCRDR,10) NACTN,(NABA(NP,J),J=NPOS,IE)
0121          IF(NACTN.LE.0) GOTO 1020
0122          IF(NPD(NP,NACTN).GT.0) GOTO 1039
0123          NPD(NP,NACTN)=NPOS
0124          GOTO 1040
0125      1039  DO 1037 K=NPOS,IF
0126          NABA(NP,K-1)=NABA(NP,K)
0127          IF(NABA(NP,K).EQ.0) GOTO 1005
0128      1037  CONTINUE
0129          GOTO 1800
0130          IE=K-1
0131      1800  NPOS=NPOS-1
0132          GOTO 1035
0133      1040  DO 1038 I=NPOS,IE
0134          IF(NABA(NP,I).EQ.0) GOTO 1036
0135      1038  CONTINUE
0136          IE=IE+1
0137          GOTO 1035
0138      1036  IE=I
0139      1035  NABA(NP,IE)=0
0140          WRITE(NPRNT,1006) NACTN,(NABA(NP,J),J=NPOS,IE)
0141          NPOS=IE+1
0142          GOTO 1026
0143      1006  FORMAT(17,2X,12(15,15),13)
0144      1020  NP=NP+1
0145          IF(NNR.GT.0.AND.NP.EQ.2) GOTO 1025
0146          IF(LOOP.GT.0.AND.NP.EQ.3) GOTO 1025
0147          DO 1041 KK=1,NOQ
0148          NFBTU(KK)=KK
0149          NABA(1,IMN)=0
0150          NABA(1,IMN-1)=0
0151          IF(NPROJ) 1591,1592,1592
0152      1591  READ(NCRDR,10) NABA(1,IMN),NABA(1,IMN-1)
0153      1592  WRITE(NPRNT,221) NOQ,NSRC,NPD,NSKST,NSKS,NPPMS,JSEED,ITIME,IBRNCH
          DATSI1590
          DATSI1591

```

FORTRAN IV G LEVEL 1, MOD 4 DATAN DATE = 70180 13/46/04 PAGE 0005

```

0154          WRITE(NPRNT,223) (NSORC(J),J=1,NSRC)
0155          WRITE(NPRNT,225) (NSINK(K),K=1,NPD)
0156          KPP=NPD+1
0157          IF(KPP.GT.NSKS) GOTO 541
0158          WRITE(NPRNT,227) (NSINK(K),K=KPP,NSKS)
0159      541  WRITE(NPRNT,229)
0160          DO 231 J=1,NOQ
0161      231  WRITE(NPRNT,233) J,NRELP(J),NTYPE(J)
0162      221  FORMAT(1H1,14X22HHIGHEST NODE NUMBER IS,15,15X,25HNUMBER OF SOURCDATSI670
          1E NODES IS,15/15X,23HNUMBER OF SINK NODES IS,15/15X,41HNUMBER OF NODATSI680
          2DES TO REALIZE THE NETWORK IS,15/15X,23HSTATISTICS COLLECTED ON,15DATSI690
          3,6H NODES/15X,27HNUMBER OF PARAMETER SETS IS,15/15X,24HINITIAL RANDATSI700
          40M NUMBER IS, 15/15X,'ANTITHETIC TIME INDICATOR IS',15/15X,
          5'ANTITHETIC BRANCH INDICATOR IS',15/)
0163      223  FORMAT(15X,19HSOURCE NODE NUMBERS/17X,10(10I5))
0164      225  FORMAT(15X,17HSINK NODE NUMBERS/17X,10(10I5))
0165      227  FORMAT(15X,34HSTATISTICS COLLECTED ALSO ON NODES/17X,10(10I5))
0166      229  FORMAT(25X,5HNODE 5X, 6HNUMBER,5X,4HNODE/34X,8HRELEASES,4X,4HTYPEDATSI750
          1/)
0167      233  FORMAT(I28,I10,I10)
0168          RETURN
0169      23  CALL ERROR(3,NSET,QSET)
0170      9999  WRITE(NPRNT,551)
0171      551  FORMAT('O ***END OF DATA***')
0172          CALLEXIT
0173          END
          DATSI1790

```

FORTRAN IV G LEVEL 1, MOD 4 DRAND DATE = 70180 13/46/04 PAGE 0001

```

0001          FUNCTION DRAND(ISEED)
0002          IF(ISEED.EQ.0) ISEED=1267
0003          ISEED=ISEED*65539
0004          IF(ISEED.GT.0) GOTO 6
0005      5  ISEED=ISEED+2147483647+1
0006      6  DRAND=ISEED
0007          DRAND=DRAND*.4656613E-9
0008          RETURN
0009          END

```

FORTRAN IV G LEVEL 1, MOD 4 ERROR DATE = 70180 13/46/04 PAGE 0001

```

0001      SUBROUTINE ERROR(J,NSET,QSET)
0002      DIMENSION NSET(1),QSET(1)
0003      COMMON IMM,PAPAM(300,4),JTRIB(4),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRI8,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,I4,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008      COMMON NTYPE(500),MX,NFTBU(500),CSTAR,CINC
0009      COMMON MNM,IMN,NPD,NOQ,NPRMS,NRUNS,NYR,MAXNS,MAXA,NPRTOP,
      INAME(6),NPROJ,MGN,NDAY,NSKST,NSRC,NRFLP(500),NRFL(500),NSKSR,
      ZNSORC(20),TOTIM,KOUNT(4),JSINK(20),NPD(3,500),NABA(3,800)
0010      GOTO (1,2,3,4,5,6,7,8,9),J
0011      1 WRITE(NPRNT,20)
0012      20 FORMAT('/// ERROR EXIT TYPE 1**EXCEEDING DIMENSIONS FOR THE NUMBER OF
      10F SOURCE NODES, THE NUMBER OF NODES ON WHICH STATISTICS ARE TO BE
      2 COLLECTED,/,/, ' THE NUMBER OF ALLOWABLE PARAMETERS, OR THE NUMBER OF
      30F ALLOWABLE NODE MODIFICATIONS')
0013      CALL EXIT
0014      3 WRITE(NPRNT,40)
0015      40 FORMAT('/// ERROR TYPE 3***THE START NODE WAS NEGATIVE, THE NUMBER
      10F NODES STATISTICS ARE TO BE COLLECTED ON IS NEGATIVE,/'
      2' OR NUMBER OF COUNTS IS NEGATIVE')
0016      GOTO 9
0017      4 WRITE(NPRNT,250)
0018      250 FORMAT('/// ERROR IN COLCT OR HISTOGRAM**THE NUMBER OF NODES STATI
      1STICS ARE TO BE COLLECTED ON IS NOT WITHIN THE ALLOWABLE RANGE')
0019      GOTO 9
0020      6 WRITE(NPRNT,60)
0021      60 FORMAT('/// THE NUMBER OF SINK NODES NEEDED TO REALIZE THE NETWORK
      11S NEGATIVE,OR THE NUMBER OF ACTIVITIES FOR A NODE IS NEGATIVE')
0022      GOTO 9
0023      8 WRITE(NPRNT,80)
0024      80 FORMAT('/// AN ELEMENT IN THE ARRAY FOR FIRST ENTRY IN THE FILE IS
      1 NEGATIVE**OCCURRED IN NFIND')
0025      GOTO 9
0026      2 WRITE(NPRNT,30)
0027      30 FORMAT('/// THE INDICATOR IN THE ACTIVITY FILE IS GREATER THAN 777
      17***ERROR NUMBER 2***OCCURRED IN FILEM OR SCHAT')
0028      GOTO 10
0029      5 WRITE(NPRNT,50)
0030      50 FORMAT('/// THE ACTIVITY FILE IS FULL**THE OVERLAP SET FOLLOWS')
0031      7 WRITE(NPRNT,70)
0032      70 FORMAT('/// THE ARRAY FOR COLLECTING STATISTICS HAS A NEGATIVE NUMBER FOR T
      1ER FOR THE COUNTER')
0033      10 WRITE(NPRNT,99)
0034      99 WRITE(NPRNT,100) J,TNOW,NRUN,MFE(1),NQ(1),IBRNCH,ITIME
0035      100 FORMAT('///36X16HERROR EXIT, TYPE,I3,7H ERROR.//21H FILE STATUS AT
      1TIME,F10.4,5X,'ON RUN NUMBER',I5,5X,'MFE(1) =',I5,' NQ(1) =',I5,

```

ERRS 10

ERRS 70

ERRS 90

FORTRAN IV G LEVEL 1, MOD 4 ERROR DATE = 70180 13/46/04 PAGE 0002

```

0036      2' IBRNCH =',I5,' ITIME =',I5/)
0037      200 WRITE(NPRNT,200)
0038      200 FORMAT(27X,'QSET',I5X,'NSET')
0039      IL=1
0040      IV=MXX
0041      DO 210 I=1,10
0042      WRITE(NPRNT,90) I,QSET(1),NSET(I),IJ=IL,IV)
0043      IL=IL+MXX
0044      210 IV=IV+MXX
0045      90 FORMAT(13X,I5,5X,E12.6,5X,11I8)
0046      WRITE(NPRNT,99)
0047      79 FORMAT(11I1)
0048      IF(NCLCT) I9,I9,18
0049      18 WRITE(NPRNT,99)
0050      98 FORMAT(/11H ARRAY SUMA,/)
0051      DO 110 I=1,NCLCT
0052      WRITE(NPRNT,81) I,(SUMA(I,K),K=1,5)
0053      81 FORMAT(110,5F10.4)
0054      WRITE(NPRNT,99)
0055      19 IF(NHIST) 9,9,12
0056      12 WRITE(NPRNT,96)
0057      96 FORMAT (/12H ARRAY JCELS/)
0058      DO 112 I=1,NHIST
0059      112 WRITE(NPRNT,26) I,(JCELS(I,K),K=1,MXC)
0060      26 FORMAT(17X,I3,5X,11I4/(25X,11I4))
0061      9 CALL GASP(NSET,QSET)
0062      GOTO 9
      END

```

ERRS 110

ERRS 130

ERRS 250

ERRS 260

ERRS 290

ERRS 300

ERRS 330

ERRS 350

ERRS 360

ERRS 380

ERRS 390

ERRS 400

ERRS 450

NOT REPRODUCIBLE

FORTTRAN IV G LEVEL 1, MOD 4 FILEM DATE = 70180 13/46/04 PAGE 0001

```

0001      SUBROUTINE FILEM (JQ,NSET,QSET)      'FILS 10
0002      DIMENSION NSET(1),QSET(1)
0003      COMMON /MM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRI8 ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,TNOM,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      IND1=MFA*MXX-1
0008      IND2= IND1+1

C
C*****TEST TO SEE IF THERE IS AN AVAILABLE COLUMN FOR STORAGE      FILS 70
C
0009      IF (MFA - ID ) 2,2,3      FILS 80
0010      3      CALL ERROR(5,NSET,QSET)
C
C*****PUT ATTRIBUTE VALUES IN FILE      FILS 120
C
0011      2      QSET(MFA)=ATRI8
0012      INDX = (MFA - 1) * MXX      FILS 200
0013      DO 10 I = 1,IM      FILS 210
0014      INDX = INDX + 1      FILS 220
0015      10 NSET(INDX) = JTRIB(I)      FILS 230
0016      MFEX=MFE(JQ)      FILS 240
0017      KNT = 2      FILS 250
0018      NXFA = NSET(IND1)      FILS 280
0019      MLEX=MLE(JQ)      FILS 290
0020      IF(MLEX)100,27,11      FILS 300
0021      100 CALL ERROR (88,NSET,QSFT)      FILS 310
0022      27 NSET(IND2) = KLE      FILS 330
0023      MFE(JQ) = MFA      FILS 340
0024      17 NSET(IND1) = KOL      FILS 360
0025      MLE(JQ) = MFA      FILS 370
0026      14 MFA =NXFA      FILS 380
0027      IF(MFA-KOF) 237,238,238      FILS 390
0028      237 NSET(NXFA*MXX)=KLE
0029      238 NQ(JQ)=NQ(JQ)+1      FILS 420
0030      RETURN      RMVS 510
0031      11 IF(JQ-1) 241,241,242      FILS 455
0032      241 IF (QSET(MFA)-QSET(MLEX)) 12,13,13
0033      242 IF (QSET(MFA)-QSET(MLEX)) 13,13,12
0034      13 IND3= MLEX*MXX-1
0035      MSU = NSET(IND3)
0036      NSET(IND3) = MFA      FILS 490
0037      NSET(IND2) = MLEX      FILS 510
0038      GO TO (18,17),KNT      FILS 520
0039      18 NSET(IND1) = MSU      FILS 540
0040      NSET(MSU*MXX)=MFA
0041      GO TO 14      FILS 570
0042      12 KNT = 1      FILS 580

```

FORTTRAN IV G LEVEL 1, MOD 4 FILEM DATE = 70180 13/46/04 PAGE 0002

```

0043      MLEX=NSET(MLEX*MXX)
0044      IF(MLEX-KLE) 11,16,11      FILS 610
0045      16 NSET(IND2)=KLE
0046      MFE(JQ) = MFA      FILS 640
0047      NSET(IND1)=MFEX
0048      NSET(MFEX*MXX)=MFA
0049      GO TO 14      FILS 690
0050      END      FILS 700

```

FORTRAN IV G LEVEL 1, MOD 4

GASP

DATE = 70180

13/46/04

PAGE 0001

```

0001      SUBROUTINE GASP (NSET,QSET)
0002      DIMENSION NSET(1),QSET(1),ATIME(2000),BRNCH(2000),ACTIME(500),
1      ISAVE(500),CSAVE(500)
0003      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MAXX
0004      COMMON MXC,ATRIB ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRNCH,ITIME,LOOP
0006      COMMON KNL,KOF,KLE
0007      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008      COMMON NTYPE(500),MX,NFTBU(500),CSTAR,CINC
0009      COMMON NNM,IMN,NPD,NQO,NPRMS,NRUNS,NYR,MAXNS,MAXA,NPRTOP ,
1      INAME(6),NPROJ,MON,NDAY,NSKST,NSRC,NRELP(500),NREL(500), NSKSR,
2      NSORC(20),TOTIM,KOUNT(4),JSINK(20),NPD(3,500),NABA(3,800)
      CALL DATAN (NSET,QSET,ATIME,BRNCH,ACTIME,LUPDO)
0010      IF(LUPDO.EQ.0) LUPDO=1
0011      DO 2125 MM4=1,LUPDO
0012      KOWT=1
0013      COSSQR=0.0
0014      CMIN=.1E20
0015      CMAX=0.0
0016      COST=0.0
0017      TOTCOS=0.0
0018      IF(NPRTOP.EQ.0) GOTO 2
0019      DO 3 I=1,NQO
0020      TSAVE(I)=0.0
0021      CSAVE(I)=0.0
0022      3      NAT=1
0023      2      IF(ITIME.EQ.0) GOTO 2168
0024      DO 6740 J=1,MAXA
0025      ATIME(J)=0.0
0026      6740 BRNCH(J)=0.0
0027      2168 IF(NCLCT.EQ.0) GOTO 2160
0028      DO 2121 I=1,NCLCT
0029      DO 2122 J=1,3
0030      2122 SUMA(I,J)=0.0
0031      SUMA(I,4)=1.0E20
0032      SUMA(I,5)=-1.0E20
0033      2121 IF(NHIST.EQ.0) GOTO 42
0034      DO 2123 K=1,NHIST
0035      DO 2123 L=1,MXC
0036      2123 JCELS(K,L)=0
0037      DO 2124 L=1,MXC
0038      2124 JCELS(20,L)=0
0039      C
      C****OBTAIN NEXT EVENT WHICH IS FIRST ENTRY IN FILE 1. ATRIB(1) IS      GSPS 110
      C****EVENT TIME, JTRIB(1) IS EVENT CODE      GSPS 120
      C
0040      42 KCOL=MFE(1)
0041      CALL RMOVE(KCOL,1,NSET,QSET,NAT,KOWT)

```

FORTRAN IV G LEVEL 1, MOD 4

GASP

DATE = 70180

13/46/04

PAGE 0002

```

0042      IF(NAT.EQ.2) GOTO 120
0043      IF(NAT.EQ.0) GOTO 120
0044      TNOW=ATRIB
0045      NEND=JTRIB(2)
      C
      C****IF END NODE IS ZERO START NETWORK BY SCHEDULING ACTIVITIES      GSPS 160
      C****FROM EACH SOURCE NODE      GSPS 170
      C
0046      IF(NEND)4,4,5      GSPS 180
0047      4 DO 10 N=1,NSRC      GSPS 190
0048      M=NSORC(N)      GSPS 200
0049      10 CALL SCHAT (M,NSET,QSET,BRNCH,ATIME)
0050      GO TO 42      GSPS 230
      C
      C****REDUCE NO. OF RELEASES FOR END NODE BY 1.      GSPS 240
      C****TEST TO SEE IF NODE IS RELEASED.      GSPS 250
      C
0051      5 NEND=NFTBU(NEND)      GSPS 260
0052      NREL(NEND)=NREL(NEND)-1      GSPS 261
0053      JA=JTRIB(1)
0054      TIME=TNOW-ACTIME(JA)
0055      COST=COST+FLOAT(JTRIB(8))+FLOAT(JTRIB(7))*TIME      GSPS 262
0056      IF(NPROJ) 511,512,512
0057      511 IF(NRUN.LT.NABA(1,IMN).OR.NRUN.GT.NABA(1,IMN-1)) GOTO 512
0058      512 WRITE(NPRNT,598) TNOW,(JTRIB(I),I=2,IM),COST
0059      598 FORMAT(5X,7HAT TIME,F8.2,17H ACTIVITY ON NODE,15,16H WITH ATTRIBUGSPS 267
      1TES,615,13H WAS REALIZED,2X,'ACCUMULATED COST =',F8.2)
      C
      C****DO NODE MODIFICATIONS OR RESET OR LOOP
      C
0060      512 IF(JTRIB(5)) 665,665,60
0061      60 L=JTRIB(5)
0062      KOUNT(L)=KOUNT(L)+1      GSPS 290
0063      665 NP=4
0064      IF(JTRIB(6).LE.0) GOTO 65
0065      JJJ=JTRIB(6)
0066      68 NP=NP-1
0067      IF(NPO(NP,JJJ).LE.0) GOTO 65
0068      NPOS=NPO(NP,JJJ)
0069      IF(NP.EQ.3) GOTO 2155
0070      69 NDN=NABA(NP,NPOS)
0071      NTBI=NABA(NP,NPOS+1)
0072      IF(NP.EQ.1) GOTO 66
0073      IF(NDN.NE.1) GOTO 2111
0074      NRELP(NDN)=NRELP(NTBI)
0075      GOTO 2010
0076      2111 NREL(NDN)=NRELP(NTBI)
0077      2010 IF(NRUN.LT.NABA(1,IMN).OR.NRUN.GT.NABA(1,IMN-1)) GOTO 1902

```

FORTRAN IV G LEVEL 1, MOD 4 GASP DATE = 70181 21/26/46 PAGE 0003

```

0078      WRITE(NPRNT,1806) NDN,NREL(NBTI)
0079      1806  FORMAT(/5X,'NODE',I5,' WAS RESET TO',I5)
0080      1902  GOTO 67
0081      2155  CALL ALLOP(NSET,QSET,NPOS,COST,ACTIME,JJJ)
0082      GOTO 68
0083      66    DO 1901 N=1,NQ
0084      IF(NFTBU(N).NE.NDN) GOTO 1901
0085      672   NFTBU(N)=NTBI
0086      669   IF(NRUN.LT.NABA(1,IMN).OR.NRUN.GT.NABA(1,IMN-1)) GOTO 1901
0087      WRITE(NPRNT,1807) NDN,NTBI,NREL(NBTI)
0088      1807  FORMAT(/5X,'NODE',I5,' WAS CHANGED TO NODE',I5,' AND SET TO',I5)
0089      1901  CONTINUE
0090      67    IF(NABA(NP,NPOS+2).LE.0) GOTO 65
0091      NPOS=NPOS+2
0092      GOTO 69
0093      65    IF(NP.EQ.2.OR.NP.EQ.3) GOTO 68
0094      IF(NREL(NEND).GT.0) GOTO 42
0095      IF(NPRTOP.EQ.0) GOTO 6
0096      IF(ACTIME(NEND).NE.0.0) GOTO 6
0097      TSAVE(NEND)=TSAVE(NEND)+TNOW
0098      CSAVE(NEND)=CSAVE(NEND)+1.0
0099      6     ACTIME(NEND)=TNOW
0100      IF(NREL(NEND).EQ.0) GOTO 7
C
C*****TEST TO SEE IF NODE CAN BE RELEASED MORE THAN ONCE.                      GSPS 400
C
0101      9   IF(NTYPE(NEND)-2)42,42,50                      GSPS 410
C
C*****TEST TO SEE IF STATISTICS ARE TO BE COLLECTED ON THE NODE.                      GSPS 420
C*****STORE TIME NODE REALIZED, SAVE TIME FIRST REALIZED
C
0102      7   DO 8 K=1,NSKS
0103      IF(NEND-NSINK(K))8,39,8                      GSPS 440
0104      39   IF(JSINK(K))50,50,40                      GSPS 450
0105      8     CONTINUE                      GSPS 460
0106      50   NT=NTYPE(NEND)
0107      GO TO (52,52,52,52,52,52,54,54),NT                      GSPS 471
0108      54   KCOL=NFIND(NEND,1,2,NSET,QSET)
0109      IF(KCOL) 52,52,56                      GSPS 473
0110      56   CALL RMOVE(KCOL,1,NSET,QSET,NAT,KOWT)
0111      GO TO 54                      GSPS 475
0112      52   CALL SCHAT (NEND,NSET,QSET,BRNCH,ATIME)
0113      GO TO (42,42,42,42,51,51,51,51),NT                      GSPS 480
0114      51   NREL(NEND)=NREL(NEND)                      GSPS 490
0115      GO TO 42                      GSPS 500
C
C*****REDUCE NO. OF SINK NODES REALIZED BY ONE. COLLECT STATISTICS.                      GSPS 510
C

```

FORTRAN IV G LEVEL 1, MOD 4 GASP DATE = 70181 21/26/46 PAGE 0004

```

0116      40  IF(K-NPD) 440,440,441                      GSPS 515
0117      440  NSKS=NSKS-1                      GSPS 520
0118      441  JSINK(K)=0                      GSPS 530
0119      CALL HISTO(TNOW,XLOW(K),WIDTH(K),K)                      GSPS 540
0120      NL=K+(K-1)*NCTS                      GSPS 550
0121      CALL COLCT (TNOW,NL)
0122      IF(NCTS) 45,45,46                      GSPS 570
0123      46   DO 70 I=1,NCTS                      GSPS 580
0124      NL=NL+1                      GSPS 590
0125      XC=KOUNT(I)                      GSPS 600
0126      CALL COLCT (XC,NL)
C
C*****TEST TO SEE IF NETWORK SIMULATION IS COMPLETE.                      GSPS 620
C
0127      45  IF(NSKS) 22,100,999                      GSPS 630
0128      999  IF(NQ(NEND))22,42,50                      GSPS 640
0129      22   CALL ERROR(6,NSET,QSET)
C
C*****NETWORK HAS BEEN SIMULATED ONE MORE TIME. ADD TNOW TO TOTAL TIME. GSPS 660
C*****ANTITHETIC TIME AND BRANCH INDICATORS
C
0130      100  TOTIM=TOTIM+TNOW                      GSPS 670
0131      IF(ETIME) 2000,2001,2002
0132      2000  ETIME=1
0133      DO 7210 I=1,MAXA
0134      7210  ATIME(I)=0.0
0135      GOTO 2001
0136      2002  ETIME=-1
0137      2001  IF(IBRNCH)2005,2006,2007
0138      2005  IBRNCH=1
0139      DO 7211 I=1,MAXA
0140      7211  BRNCH(I)=0.0
0141      GOTO 2006
0142      2007  IBRNCH=-1
0143      2006  CONTINUE
C
C*****REINITIAL FOR ANOTHER RUN BY REMOVING ALL EVENTS FROM EVENT                      GSPS 700
C*****FILE AND RESETING NETWORK VALUES.                      GSPS 710
C
0144      115  IF(NQ(I))22,120,116                      GSPS 720
0145      116   KCOL=MFE(I)
0146      CALL RMOVE(KCOL,1,NSET,QSET,NAT,KOWT)
0147      IF(NAT.EQ.2) GOTO 120
0148      IF(NAT.EQ.0) GOTO 120
0149      GO TO 115                      GSPS 740
0150      120   JTRIB(2)=0
0151      ATRIB=0.0
0152      CALL FILEM(1,NSET,QSET)                      GSPS 780

```

FORTRAN IV G LEVEL 1, MOD 4

GASP

DATE = 70180

13/46/04

PAGE 0005

```

C*****STORE COST INFORMATION
C
0152      CALL HISTO (COST,CSTAR,CINC,2)
0153      COSSQR=COSSQR+COST*COST
0154      CMIN=AMINI(CMIN,COST)
0155      CMAX=AMAX1(CMAX,COST)
0156      TOTCOS=TOTCOS+COST
0157      COST=0.0
0158      TNOW=0.0
0159      NSKSR=NSKST
0160      DO 80 K=1,NSKS
0161      80 JSINK(K)=1
0162      DO 85 J=1,NCTS
0163      85 KOUNT(J)=0
0164      DO 37 IJ=1,NQQ
0165      ACTIME(IJ)=0.0
0166      NFBUI(IJ)=IJ
0167      37 NRFL(IJ)=NRFLP(IJ)
C
C*****TEST TO SEE IF ALL RUNS HAVE BEEN MADE.
C
0168      IF(NAT.EQ.0) GOTO 2126
0169      IF(NRUNS-NRUN) 110,110,6718
0170      6718 NRUN=NRUN+1
0171      NAT=1
0172      ISAVE=ISEED
0173      GOTO 42
0174      110 CALL SUMRY (NSET,TSAVE,CSAVE,TOTCOS,COSSQR,CMIN,CMAX)
0175      NRUN=1
0176      GOTO 2125
0177      2126 ISEED=ISAVE
0178      WRITE(NPRNT,2130) NRUN
0179      2130 FORMAT(10X,' RUN NUMBER IS',
0180      NAT=-1
0181      KOWT=KOWT+1
0182      NABA(1,IMN)=NRUN
0183      NABA(1,IMN-1)=NPRUN
0184      IF(itime.LE.0) GOTO 2163
0185      DO 2164 I=1,MAXA
0186      2164 ATIME(I)=0.0
0187      2163 IF(ibrnch.LE.0) GOTO 2166
0188      DO 2167 I=1,MAXA
0189      2167 BRNCH(I)=0.0
0190      2166 IF(NPROJ.LE.0) GOTO 42
0191      NPROJ=-NPROJ
0192      GOTO 42
0193      2125 CONTINUE
0194      RETURN

```

GSPS 790

GSPS 800

GSPS 810

GSPS 820

GSPS 830

GSPS 840

GSPS 860

GSPS 970

GSPS 680

GSPS 950

FORTRAN IV G LEVEL 1, MOD 4

GASP

DATE = 70180

13/46/04

PAGE 0006

```

0105      END

```

GSPS 960

FORTRAN IV G LEVEL 1, MOD 4

HISTO

DATE = 70180

13/46/04

PAGE 0001

```

0001      SUBROUTINE HISTO (X1,4,W,N)
0002      COMMON IMM,PAPAM(300,4),JTRIR(4),ISEED,MFE(500),NPRNT,MXX
0003      COMMON MXC,ATPIS ,JCFLS(20,37),SUMA(100,5),NHIST,NCRDR
0004      IF(W.EQ.0.0) RETURN
0005      IF(N.EQ.20) GOTO 3
0006      IF(N.LE.NHIST.AND.N.GT.0) GOTO 3
0007      CALL ERROR(4,NSFT,QSET)
C
C*****TRANSLATE X1 BY SUBTRACTING A IF X.LE.A THEN ADD 1 TO FIRST CELL
C
0008      3 X = X1 - A
0009      IF (X)6,7,7
0010      6 IC = 1
0011      GO TO 8
C
C*****DETERMINE CELL NUMBER IC. ADD 1 FOR LOWER LIMIT CELL AND 1 FOR
C*****TRUNCATION
C
0012      7 IC = X/W + 2. +.0001
0013      IF(IC.GT.MXC) IC=MXC
0014      8 JCELS(N,IC) = JCELS(N,IC) + 1
0015      RETURN
0016      END

```

HSTS 10

HSTS 110

HSTS 120

HSTS 130

HSTS 140

HSTS 150

HSTS 160

HSTS 170

HSTS 180

HSTS 210

HSTS 220

HSTS 230

```

FORTRAN IV G LEVEL 1, MOD 4      NFIND      DATE = 70180      13/46/04      PAGE 0001

0001      FUNCTION NFIND(INVAL,J0,JATT,NSET,QSET)      FIND 10
0002      DIMENSION NSET(1),QSET(1)
0003      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      NEXTK=MFE(J0)      FIND 70
0005      IF(NEXTK) 16,1,20      FIND 80
0006      16 CALL ERROR(8,NSET,QSET)
0007      1 NFIND=0      FIND 100
0008      RETURN      FIND 110
0009      20 INDX=(NEXTK-1)*MXX +JATT      FIND 120
0010      IF(NSET(INDX)-NVAL) 4,21,4      FIND 130
0011      4 NEXTK=NSET(NEXTK*MXX-1)
0012      IF(NEXTK-7777) 20,1,1      FIND 160
0013      21 NFIND=NEXTK      FIND 170
0014      RETURN      FIND 180
0015      END      FIND 190

```

```

FORTRAN IV G LEVEL 1, MOD 4      RMOVE      DATE = 70180      13/46/04      PAGE 0001

0001      SUBROUTINE RMOVE (KCOL,J0,NSET,QSET,NAT,KONT)
0002      DIMENSION NSET(1),QSET(1)
0003      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATTRIB ,JCFLS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,INOW,MLE(500),NQ(500),NNK,IBRNCH,ITIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      IF (KCOL) 16,16,2      RMVS 150
0008      16 IF(KONT.GE.5) GOTO 1000
0009      IF(NAT.EQ.1) GOTO 50
0010      NAT=2
0011      RETURN
0012      50 NAT=0
0013      SETUPN
0014      1000 CALL ERROR(9,NSET,QSET)
0015      C *****PUT VALUES OF KCOL IN ATTRIB,JTRIB      RMVS 170
0016      C
0017      2 ATTRIB=QSET(KCOL)
0018      QSET(KCOL)=0.0
0019      INDX = (KCOL - 1) * MXX      RMVS 220
0020      DO 10 I =1,IM      RMVS 230
0021      INDX = INDX + 1      RMVS 240
0022      JTRIB(I) = NSET(INDX)      RMVS 250
0023      10 NSET(INDX) = 0      RMVS 330
0024      INDX=KCOL*MXX      RMVS 340
0025      JL = NSET(INDX - 1)      RMVS 350
0026      JK = NSET(INDX)      RMVS 360
0027      INO1=JK*MXX-1
0028      INO2=JL*MXX
0029      IF (JL- KOL) 33,34,33      RMVS 370
0030      33 IF (JK- KLF) 35,36,35      RMVS 380
0031      35 NSET(INO1)=JL
0032      NSET(INO2)=JK
0033      INDX=KCOL*MXX-1      RMVS 430
0034      NSET(INDX) = MFA      RMVS 440
0035      NSET(INDX+1) = KLE      RMVS 450
0036      IF(MFA-KOF) 234,235,235      RMVS 460
0037      234 NSET(MFA*MXX)=KCOL
0038      235 MFA=KCOL      RMVS 490
0039      NQ(J0) = NQ(J0)-1      RMVS 500
0040      RETURN
0041      36 NSET(INO2)=KLE      RMVS 540
0042      MFE(J0) = JL      RMVS 550
0043      GO TO 37      RMVS 560
0044      34 IF (JK-KLE) 38,39,38
0045      38 NSET(INO1)=KOL      RMVS 590
0046      MLE(J0) = JK      RMVS 600
0047      GO TO 37

```

```

FORTRAN IV G LEVEL 1, MOD 4      REMOVE      DATE = 70180      13/46/04      PAGE 0002

0046      39 MFE(J0) = 0      RMVS 610
0047      MLE(J0) = 0      RMVS 620
0048      GO TO 37      RMVS 630
0049      END      RMVS 640

```

```

FORTRAN IV G LEVEL 1, MOD 4      RNORM      DATE = 70180      13/46/04      PAGE 0001

0001      FUNCTION RNORM (J,RA,RB)
0002      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0003      V=(-2.0*ALOG(RA))*0.5*COS (6.283*RB)      NRLS 80
0004      RNORM = V*PARAM (J,4) + PARAM (J,1)      NRLS 90
0005      IF (RNORM-PARAM (J,2)) 6,7,8      NRLS 100
0006      6 RNORM = PARAM (J,2)      NRLS 110
0007      7 RETURN      NRLS 120
0008      8 IF (RNORM-PARAM (J,3)) 7,7,9      NRLS 130
0009      9 RNORM = PARAM (J,3)      NRLS 140
0010      RETURN      NRLS 150
0011      END      NRLS 160

```

FORTPAN IV G LEVEL 1, MOD 4 SAMPL DATE = 70180 13/46/04 PAGE 0001

```

0001      SUBROUTINE SAMPL(DEV,ATIME,NEXT)
0002      DIMENSION ATIME(1)
0003      COMMON IMM,PARAM(300,4),JTRIS(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRIB ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON IO,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,IBRANCH,ITIME,LOOP
0006      J=JTRIS(3)
0007      JD=JTRIS(4)
0008      IF(JD.EQ.1) GOTO 1

C
C*****JD=JTRIS(4)=1 CONSTANT
C*****      2 NORMAL
C*****      3 UNIFORM
C*****      4 ERLANG
C*****      5 LOGNORMAL
C*****      6 POISSON
C**** IF ITIME IS NEGATIVE USE PREVIOUSLY GENERATED TIME
C**** IF ITIME IS ZERO RUN THE PROGRAM NORMALLY
C**** IF ITIME IS POSITIVE GENERATE TIMES
C
0009      IF(ITIME) 50,60,70
0010      60 JD=JD+5
0011      70 GO TO (1,2,3,4,2,6,52,53,4,52, 6) ,JD
0012      1 DEV=PARAM(J,1)
0013      RETURN
0014      2 RA=ORAND(ISEED)
0015      RB=ORAND(ISEED)
0016      DEV=RNDORM(J,RA,RB)
0017      79 IF(JD.EQ.5) GOTO 81
0018      TIBR=2.*PARAM(J,1)-DEV
0019      CALL ANTIM (1,NEXT,ATIME,TIBR)
0020      GOTO 77
0021      52 RA=OPAND(ISEED)
0022      RB=ORAND(ISEED)
0023      DEV=RNDORM(J,RA,RB)
0024      IF(JD.EQ.10) GOTO 82
0025      77 RETURN
0026      81 TIBR=EXP(2.*PARAM(J,1)-DEV)
0027      CALL ANTIM (1,NEXT,ATIME,TIBR)
0028      82 DEV=EXP(DEV)
0029      RETURN
0030      3 RNUM=ORAND(ISEED)
0031      TIBR=PARAM(J,2)+(PARAM(J,3)-PARAM(J,2))*(1.0-RNUM)
0032      CALL ANTIM (1,NEXT,ATIME,TIBR)
0033      GOTO 79
0034      53 RNUM=OPAND(ISEED)
0035      78 DEV=PARAM(J,2)+(PARAM(J,3)-PARAM(J,2))*RNUM
0036      RETURN
0037      4 K=PARAM(J,4)

```

NOT REPRODUCIBLE

FORTPAN IV G LEVEL 1, MOD 4 SAMPL DATE = 70180 13/46/04 PAGE 0002

```

0038      IF(K-1) 118,10,10
0039      118 WRITE(NPRNT,20)J
0040      20 FORMAT(/' K=0 FOR ERLANG',I7)
0041      CALLEXIT
0042      10 DEV=0.0
0043      ADEV=0.0
0044      DD112 I=1,K
0045      RNUM=OPAND(ISEED)
0046      IF(ITIME.EQ.0) GOTO 112
0047      ADEV=ADEV+PARAM(J,1)*ALOG(1.0-RNUM)
0048      DEV=DEV+PARAM(J,1)*ALOG(RNUM)
0049      IF(ITIME.LE.0) GOTO 7757
0050      IF(ADEV+PARAM(J,2)) 217,215,216
0051      217 TIBR=PARAM(J,2)
0052      GOTO 54
0053      215 TIBR=ADEV
0054      GOTO 54
0055      216 IF(ADEV.LE.PARAM(J,3)) GOTO 215
0056      TIBR=PARAM(J,3)
0057      CALL ANTIM (1,NEXT,ATIME,TIBR)
0058      7757 IF(DEV+PARAM(J,2)) 117,115,116
0059      117 DEV=PARAM(J,2)
0060      115 RETURN
0061      116 IF(DEV.LE.PARAM(J,3)) GOTO 115
0062      DEV=PARAM(J,3)
0063      RETURN
0064      6 NP=0
0065      NPSSN=0
0066      P=PARAM(J,1)
0067      IF(P.GT.6.) GOTO 14
0068      Y=EXP(-P)
0069      X=ORAND(ISEED)
0070      Z=1.0-X
0071      13 RNUM=ORAND(ISEED)
0072      IF(ITIME.LE.0) GOTO 91
0073      IF(X.GE.Y.AND.Z.GE.Y) GOTO 8001
0074      IF(Z.GE.Y) GOTO 8003
0075      91 IF(X.GE.Y) GOTO 8002
0076      GOTO 16
0077      8001 X=X*RNUM
0078      NPSSN=NPSSN+1
0079      8003 Z=Z*(1.0-RNUM)
0080      NP=NP+1
0081      GOTO 13
0082      8002 NPSSN=NPSSN+1
0083      X=X*RNUM
0084      GOTO 13
0085      14 TEMP=PARAM(J,4)

```

```

FORTRAN IV G LEVEL 1, MOD 4          SAMPL          DATE = 70180          13/46/04          PAGE 0003

0096      PARAM(J,4)=(PARAM(J,1))*0.5
0097      RA= DRAND(ISEED)
0098      RB=DRAND(ISEED)
0099      DE=RNDRM(J,RA,RB)
0090      IF(ETIME.LE.0) GOTO 24
0091      NP=2.*P-DE +0.5
0092      24 NPSSN=DE +0.5
0093      16 IF(ETIME.LE.0) GOTO 23
0094      NP=PARAM(J,2)+NP
0095      IF(NP-PARAM(J,3)) 21,21,22
0096      22 NP=PARAM(J,3)
0097      21 TIBR=NP
0098      CALL ANTIM (1,NEXT,ETIME,TIBR)
0099      23 NPSSN=PARAM(J,2)+NPSSN
0100      IF(NPSSN-PARAM(J,3)) 17,17,19
0101      19 NPSSN=PARAM(J,3)
0102      17 DEV=NPSSN
0103      RETURN
0104      50 CALL ANTIM (2,NEXT,ETIME,TIBR)
0105      IF(TIBR.EQ.0.0) GOTO 60
0106      DEV=TIBR
0107      RETURN
0108      END

```

```

FORTRAN IV G LEVEL 1, MOD 4          SCHAT          DATE = 70180          13/46/04          PAGE 0001

0001      SUBROUTINE SCHAT (NODE,NSET,QSET,BRNCH,ETIME)
0002      DIMENSION NSET(1),QSET(1),ETIME(1),BRNCH(1)
0003      COMMON IM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRI8 ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NO(500),NNR,IBRNCH,ETIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008      COMMON NTYPE(500),MX,NFTBU(500),CSTAR,CINC
0009
0010      C*****NEXT IS LOCATION OF FIRST ENTRY IN FILE OF ACTIVITIES WITH START
0011      C*****NODE- NODE. NT IS THE NODE TYPE.
0012      C
0013      NDT=NFTBU(NODE)
0014      NT=NTYPE(NODE)
0015      NEXT=MFE(NODE)
0016      IF(NEXT) 4,4,5
0017      C
0018      C*****GENERATE A RANDOM NUMBER RNUM.
0019      C
0020      5 RNUM=DRAND(ISEED)
0021      HOLD=RNUM
0022      C
0023      C*****OBTAIN STARTING LOCATION IN NSET OF ENTRY.
0024      C
0025      1 INDXN=(NEXT-1)*MXX
0026      C*****TEST IF DETERMINISTIC OR PROBABILISTIC NODE.
0027      C
0028      GO TO (100,200,100,200,100,200,100,200),NT
0029      100 DO 10 I=1,IM
0030      INDX=INDXN+I
0031      10 JTRIB(I)=NSET(INDX)
0032      C
0033      C*****OBTAIN SAMPLE FOR ACTIVITY.
0034      C
0035      CALL SAMPL (DEV,ETIME,NEXT)
0036      ATRI8=TNOW+DEV
0037      C
0038      C*****FILE END OF ACTIVITY EVENT IN EVENT FILE.
0039      C
0040      CALL FILEM(1,NSET,QSET)
0041      GO TO (110,4,110,4,110,4,110,4),NT
0042      C
0043      C*****DETERMINE IF OTHER ACTIVITIES ARE IN FILE.
0044      C
0045      110 INDX=INDXN+MX
0046      NEXT=NSET(INDX)
0047      IF(NEXT-7777)1,4,23
0048
0049      SCAT 100
0050      SCAT 110
0051      SCAT 120
0052      SCAT 130
0053      SCAT 140
0054      SCAT 150
0055      SCAT 160
0056      SCAT 170
0057      SCAT 180
0058      SCAT 190
0059      SCAT 200
0060      SCAT 210
0061      SCAT 220
0062      SCAT 230
0063      SCAT 240
0064      SCAT 250
0065      SCAT 280
0066      SCAT 290
0067      SCAT 300
0068      SCAT 310
0069      SCAT 320
0070      SCAT 330
0071      SCAT 340

```

```

FORTRAN IV G LEVEL 1, MOD 4          SCHAT          DATE = 70180          13/46/04          PAGE 0002

0027      23 CALL ERROR(2,NSET,QSET)
0028      4 RETURN
0029      C
0030      C*****TEST RNUM AGAINST PROBABILITY (CUM.) OF REALIZING THE ACTIVITY.
0031      C*****ANTITHETIC BRANCHING
0032      C
0033      200 IF(IBRNCH) 201,202,203
0034      201 CALL ANTIM (2,NEXT,BRNCH,TIBR)
0035      RNUM=1.0-TIBR
0036      IF(TIBR.EQ.0.0) RNUM=HOLD
0037      GOTO 202
0038      203 CALL ANTIM (1,NEXT,BRNCH,PNUM)
0039      202 IF(QSET(NEXT)-RNUM) 110,110,100
0040      END
0041      SCAT 360
0042      SCAT 390

```

FORTRAN IV G LEVEL 1, MOD 4

SUMRY

DATE = 70189

21/02/42

PAGE 0001

```

0001      SUBROUTINE SUMRY (NSET,TSAVE,CSAVE,TOTCOS,COSSQR,CMIN,CMAX)
0002      DIMENSION NSET(1),TSAVE(1),CSAVE(1)
0003      COMMON IMM,PARAM(300,4),JTRIB(8),ISEED,MFE(500),NPRNT,MXX
0004      COMMON MXC,ATRIB ,JCELS(20,32),SUMA(100,5),NHIST,NCRDR
0005      COMMON ID,IM,NCLCT,MFA,TNOW,MLE(500),NQ(500),NNR,TBRACH,ITIME,LOOP
0006      COMMON KOL,KOF,KLE
0007      COMMON NRUN,XLOW(20),NCTS,NSINK(20),WIDTH(20),NSKS
0008      COMMON NTYPE(500),MX,NFTBU(500),CSTAR,CINC
0009      COMMON NNM,IMN,NPD,NQ,NPRMS,NRUNS,NYR,MAXNS,MAXA,NPRTOP ,
      1NAME(6),NPROJ,MON,NDAY,NSKST,NSRC,NREL(500),NREL(500), NSKSR,
      2NSORC(20),TOTIM,KOUNT(4),JSINK(20),NPO(3,500),NABA(3,800)

0010      21 FORMAT(1H1) SMYS 100
0011      10 FORMAT(32X,23HGERT SIMULATION PROJECT,I4,4H BY ,6A2/45X,4HDATE,I3,SMYS 110
      11H/,I3,1H/,I5//) SMYS 120
0012      15 FORMAT(/31X19H**FINAL RESULTS FOR,I5,14H SIMULATION**/) SMYS 130
0013      16 FORMAT(16X4HNODE3X11HPRB./COUNT,5X4HMEAN,4X4HSTD.DEV.,4X, SMYS 140
      14HMIN.,6X,MAX.,4X,STD.BR OF MEAN')
0014      18 FORMAT(/43X,14H**HISTOGRAMS**//20X ,5HLOWEP,3X,4HCELL/13XSMYS 160
      1,4HNODE,3X,5HLIMIT,3X,5HWIDTH,31X,11HFREQUENCIES ) SMYS 170
0015      19 FORMAT(/116,F9.2,F8.2,5X,1116/(38X,1116)) SMYS 180
0016      WRITE(NPRNT,21) SMYS 200
0017      WRITE (NPRNT,10) NPROJ,NAME,MON,NDAY,NYR SMYS 210
0018      WRITE (NPRNT,15) NRUN SMYS 220
0019      WRITE(NPRNT,16) SMYS 230
0020      NCTS1=NCTS+1 SMYS 240
0021      DO 6 K=1,NSKS SMYS 250
0022      DO 6 J=1,NCTS1 SMYS 260
0023      I=(K-1)*NCTS1+J SMYS 270
0024      JJ=J-1 SMYS 280
0025      IF(SUMA(I,3))5,62,61 SMYS 290
0026      5 CALL EPQQR(7,NSET,QSET)
0027      62 WRITE(NPRNT,63) NSINK(K),JJ SMYS 320
0028      63 FORMAT(I19,I13,OX,18HNO VALUES RECORDED) SMYS 330
0029      GO TO 6
0030      61 AVG=SUMA(I,1)/SUMA(I,3)
0031      IF(SUMA(I,3).LE.1.0) GOTO 44
0032      STD=((SUMA(I,3)*SUMA(I,2))-SUMA(I,1)*SUMA(I,1))/
      A(SUMA(I,3)*(SUMA(I,3)-1.0))**.5
      GOTO 43
0033      44 STD=0.0
0034      43 IF(JJ) 41,41,42
0035      41 PROB=SUMA(I,3) / FLOAT(NRUN)
0036      STMER=STD/SQRT(PROB*FLOAT(NRUN))
0037      WRITE(NPRNT,17) NSINK(K),PROB,AVG,STD,SUMA(I,4),SUMA(I,5),STMER SMYS 420
0038      17 FORMAT(/118,F9.4,7X,4F10.4,F15.4)
0039      GO TO 6
0040      42 WRITE(NPRNT,20) NSINK(K),JJ,AVG,STD,SUMA(I,4),SUMA(I,5) SMYS 440
0041      20 FORMAT(I18,I13,3X,4F10.4) SMYS 450
0042      6 CONTINUE SMYS 460
0043      WRITE(NPRNT,19) SMYS 470
0044      DO 82 I=1,NHIST SMYS 480
0045      82 WRITE(NPRNT,19) NSINK(I),XLOW(I),WIDTH(I),(JCELS(I,J),J=1,MXC) SMYS 490
0046      WRITE(NPRNT,21) SMYS 510
0047      WRITE(NPRNT,18) SMYS 530
0048      WRITE(NPRNT,6735) CSTAR,CINC,(JCELS(20,J),J=1,MXC)
0049      6735 FORMAT(/12X,'COST',F9.2,F9.2,5X,1116/(38X,1116))
0050      6719 FORMAT(' AVE. COST PER REALIZATION',F9.2,' STANDARD DEVIATION',
      A6X,F8.2,' MIN. COST FOR REALIZATION',F9.2,' MAX. COST FOR REALIZAT
      BION',F8.2//)
0051      IF(NRUN.EQ.1) GOTO 22
      CSTD=((FLOAT(NRUN)*COSSQR-TOTCOS)/(FLOAT(NRUN*(NRUN-1))
      A))**.5
      GOTO 23
0052      22 CSTD=0.0
0053      23 TOTCOS=TOTCOS/FLOAT(NRUN)
      WRITE(NPRNT,6719) TOTCOS ,CSTD,CMIN,CMAX
0054      IF(NPRTOP.EQ.0) GOTO 100
0055      DO 6720 I=1,NQ
0056      IF(CSAVE(I).EQ.0.0) GOTO 6720
0057      TSAVE(I)=TSAVE(I)/CSAVE(I)
0058      6720 CONTINUE
0059      WRITE(NPRNT,6717)
0060      6717 FOPHAT(4X,'NODE START',5X,'END',12X,'WITH ATTRIBUTES',
      A78S,'TIME/END NODE',2X,'TIME/ACTIVITY',7X,'PERCENT')
      DN 6715 JC=2,NQ
      LINE=MFE(JC)
      IF(LINE.EQ.0) GOTO 6715
0061      6713 IB=(LINE-1)*MXX+1
      TAVG=0.0
      CAVG=0.0
      IN=NSET(1B+1)
      IE=1B+IM-1
      IF(JC.EQ.IN) GOTO 6740
      IF(TSAVE(JC).NE.0.0) GOTO 6730
      DO 6731 I=1,NSPC
      IF(JC.EQ.NSORC(I)) GOTO 6730
0062      6731 CONTINUE
      GOTO 6736
0063      6730 IF(TSAVE(IN).EQ.0.0) GOTO 6721
      TAVG=TSAVE(IN)-TSAVE(JC)
      CAVG=TSAVE(IN)
      PERC=CSAVE(IN)/FLOAT(NRUN)
0064      6721 WRITE(NPRNT,6714) (NSET(I),I=1B,IE),CAVG,TAVG,PERC
0065      6714 FORMAT(2X,8I10,F10.2,6X,F10.2,8X,F10.2)
      GOTO 6750
0066      6740 WRITE (NPRNT,6745) (NSET(I),I=1B,IE)
0067      6745 FORMAT(2X,8I10,4X,'NO INFORMATION')
0068      6750 LINE=NSET(IE+1)
      IF(LINE.LT.7777) GOTO 6713
0069      6715 CONTINUE
0070      100 RETURN
0071      END

```

NOT REPRODUCIBLE

SMYS 550

Variance Reduction in the GERT II Simulator

One of the most promising approaches to variance reduction in connection with the GERT-II Simulator is through the introduction of an antithetic variable. The theory of this technique is based on the linear estimate of a parameter, say t , given in Equation (1).

$$(1) \quad \hat{t} = a_1 \hat{t}_1 + a_2 \hat{t}_2$$

Here, a_1 and a_2 are arbitrary constants, and $\hat{t}_1$ is a "regular" estimator of t while $\hat{t}_2$ is the antithetic variable estimator. Now, $\hat{t}$ is an unbiased estimate of t if $a_1 + a_2 = 1$, and $E(\hat{t}_1) = E(\hat{t}_2) = t$. Also, the variance of this estimate is given by Equation (2).

$$(2) \quad V(\hat{t}) = a_1^2 \sigma_1^2 + 2\rho a_1 a_2 \sigma_1 \sigma_2 + a_2^2 \sigma_2^2$$

Here, σ_1^2 and σ_2^2 are the variances of $\hat{t}_1$ and $\hat{t}_2$ respectively, and ρ is the correlation coefficient between $\hat{t}_1$ and $\hat{t}_2$. If we let $a_1 = a_2 = 1/2$, and if $\sigma_1 = \sigma_2 = \sigma$, which is the case in the procedure being described here, then equation (2) reduces to (3).

$$(3) \quad V(\hat{t}) = \sigma^2(1 + \rho)/2.$$

Thus, the objective of the use of the antithetic variable is to achieve as strong a negative correlation as possible between the estimators t_1 and t_2 , which makes $V(\hat{t})$ approach zero as ρ approaches minus one. To achieve this, two approaches are possible in the GERT-II Simulator;

- 1) Through the choice of random values for the activity duration times, and
- 2) Through the choice of random numbers used to determine branching from the stochastic nodes.

In each of these cases, the desired result can be obtained by preserving the order of the random numbers used in the simulation so that wherever the random number r_i is used in the estimator $\hat{t}_1$, the number $(1 - r_i)$ will be used in the estimator $\hat{t}_2$. In this way, pairs of random samples from any symmetrical distribution will have a correlation of -1. The effect in branching, however, is not quite so clear.

The modification of GERT II for the above purpose was accomplished by storing in an array (parallel in form to Qset and Nset) up to a maximum of four each, the values of antithetic random times and branching numbers for each activity and node completed during odd numbered realizations of the network being studied.* If feedback was present and a certain activity was completed more than once, a second value was placed in storage for both time and branching. Fifth and subsequent realizations of the same activity were not stored. Next, during even numbered realizations the stored antithetic activity duration and/or branching numbers were used in the same order in which they were placed in storage. If no stored variables were available at any time, new ones were generated as needed.

*Special procedures are sometimes required, e.g. the usual procedure for generating normally distributed random variates requires a pair of random numbers, and it will not produce a negative correlation if the above procedure is followed.

To illustrate this procedure, the PERT network shown in Figure II-1 was studied with and without the use of the antithetic variable. In this study a total of 20 simulation runs were made, where each run consisted of 250 realizations of the network. Statistics on the times required to realize nodes 5, 6, 7, 8 and 9 are given in Table II-1.

First, in the top half of the table the grand averages of the two sets of 10 runs each, and the variance among the 10 run means are given for the simulation with and without the antithetic variable. For all nodes the two grand means are almost identical and equal to the theoretical values, thus indicating both simulation procedures are unbiased. However, there is a considerable difference in the variance among the run means. The ratio of these variances indicates the effective increase in sample size due to the use of the antithetic variable. The ratio is infinite for nodes 5, 6, and 7 since the occurrence time of these nodes is merely the direct sum of one or two normally distributed random variates, and perfect negative correlation is achieved for the antithetic variable.

Examination of the lower portion of Table II-1 indicates that the use of the antithetic variable actually represents a trade-off, because estimates of the standard deviation of the actual times to the realization of a particular node are poorer when the antithetic variable is used. This results because the number of independent random samples is reduced by one-half when the antithetic variable is used. The

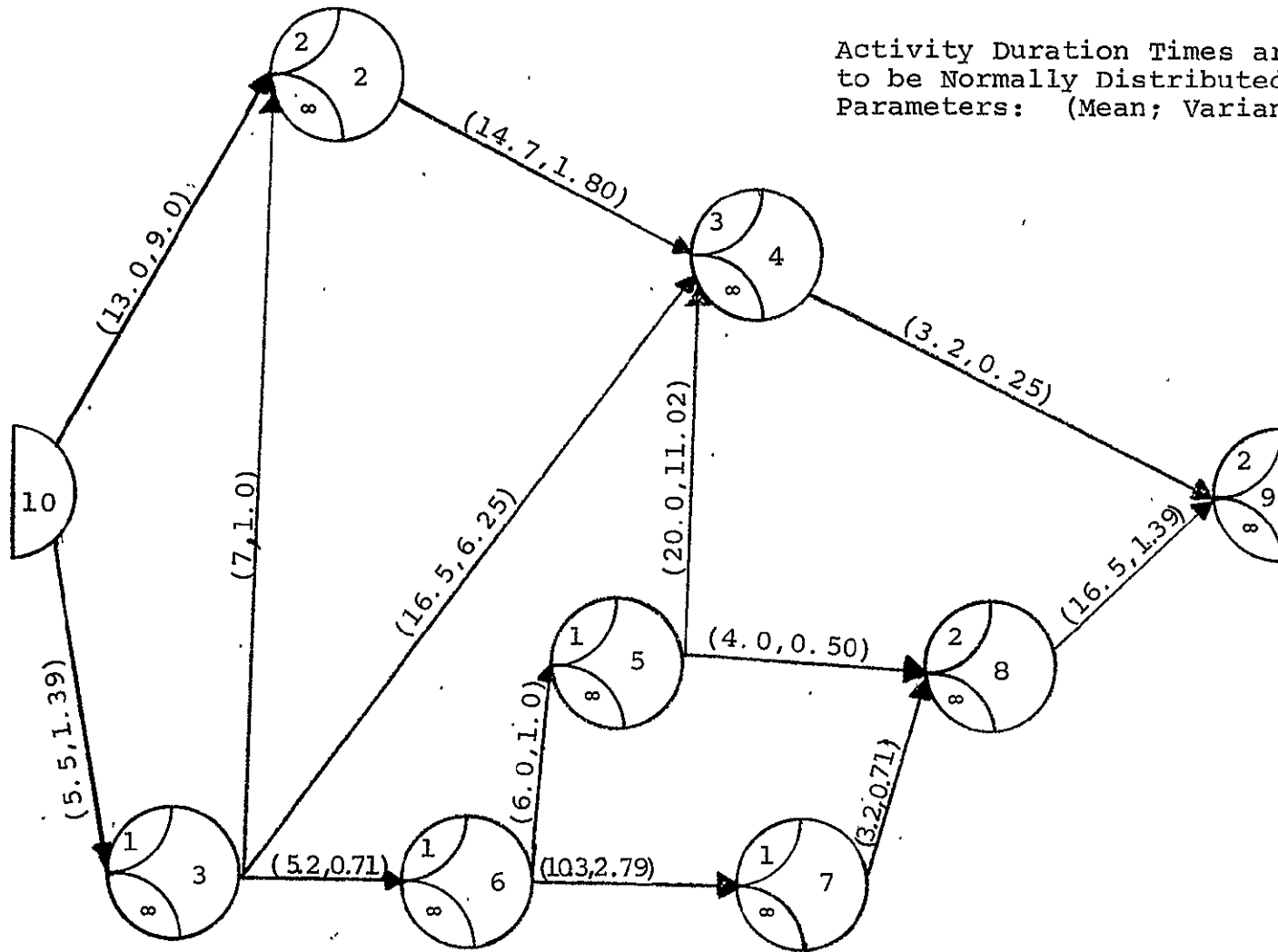


Figure II-1. Illustrative PERT Network (Project 13 in Reference 1).

variance ratios given in the last column vary randomly about the expected value of two, and the average ratio of 2.09 is quite close to this value.

To illustrate the effect of antithetic branching as well as activity duration times, the network shown in Figure II-2 was used. A procedure similar to that used for Figure II-1 was employed. A total of 80 simulation runs were made, 20 for each of the four conditions listed below, where each run consisted of 250 realizations of the network.

- 1) NONE: no antithetic variables
- 2) Branching: antithetic branching only
- 3) Time: antithetic activity duration times only
- 4) B and T: antithetic branching and activity duration times.

The above experiment was carried out using exponentially distributed activity duration times, and then the entire experiment was repeated using normally distributed times. The results of these two experiments, given in Table II-2, are discouraging. While the four procedures studied appear to be unbiased, there is no statistically significant evidence of a reduction in the variance among the run estimates.

The results for the B and T procedure were slightly superior to the average results in most cases in Table II-2. Further testing of this procedure was carried out on the large network shown in Figure 6. The results were again negative.

Table II-1. Results of the Simulation of the Network Given in Figure II-1.

Node No.	Statistic	Grand Ave. of Run Estimates			Variance Among Run Estimates		
		With	Without	True Est.	With	Without	Ratio*
6	Means	10.700	10.711	10.700	Zero	.00578	∞
5	"	16.700	16.696	16.700	Zero	.01150	∞
7	"	21.000	21.022	21.000	Zero	.00864	∞
8	"	24.256	24.270	-	.00017	.01900	112.0
9	"	42.021	42.027	-	.0185	.0236	1.28
6	Std. Dev.	1.419	1.437	1.448	.00720	.00406	1.77
5	" "	1.790	1.733	1.760	.01470	.00756	1.94
7	" "	2.202	2.187	2.211	.03080	.00884	3.48
8	" "	2.270	2.267	-	.02560	.01070	2.39
9	" "	2.705	2.708	-	.01730	.01960	0.88
						Average	2.09

*Without/with for Means and with/without for Std. Dev.

Conclusion

The results of these tests suggest that the antithetic variable is useful on PERT type networks if it is desirable to improve the estimate of the mean at the expense of the variance estimate. In general GERT type networks, however, it is not advisable to use the antithetic variable as employed in this study. More involved procedures on the use of the antithetic variable may, however, improve its effectiveness.

Each Activity Duration is Assumed to be a Random Variable with Parameters
(Probability, Mean, Variance).

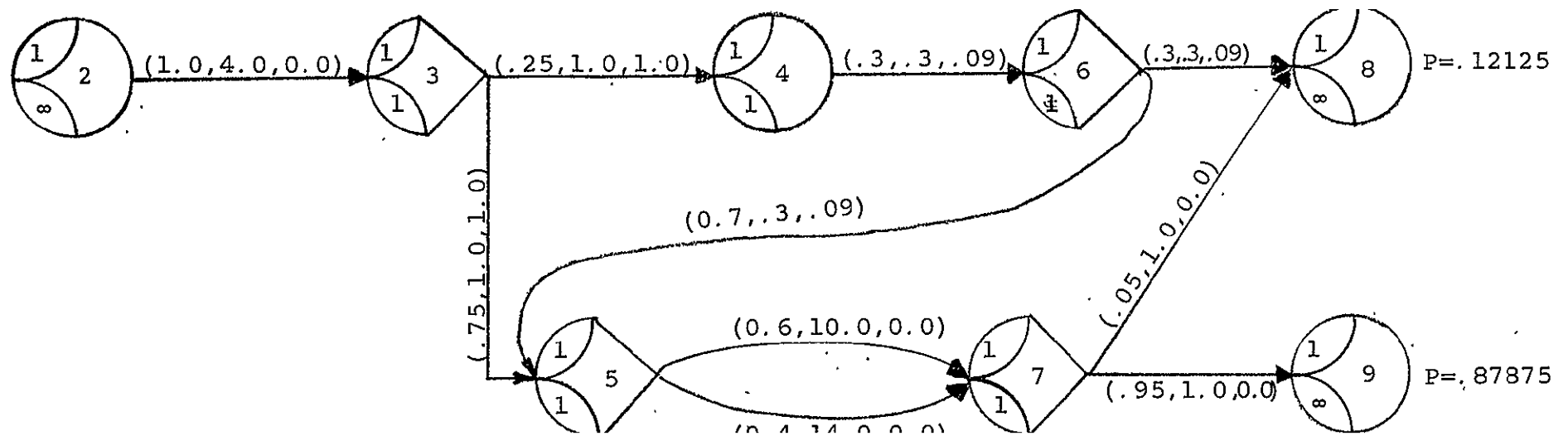


Figure II-2. GERT Network of a Manufacturing Process (Project 3 in Reference 1).

Table II-2. Results of the Simulation of the Network Given in Figure II-2.

Variance Reduction Technique	Statistic	EXPONENTIAL ACTIVITY DURATIONS				NORMAL ACTIVITY DURATIONS			
		Grand Average of Run Estimates		Variance Among Run Estimates		Grand Average of Run Estimates		Variance Among Run Estimates	
		Node 8	Node 9	Node 8	Node 9	Node 8	Node 9	Node 8	Node 9
None	Mean Value	12.16	18.26	.660	.0228	16.19	21.54	.641	.0349
Branching	" "	12.14	18.22	.564	.0312	16.15	21.55	1.063	.0375
Time	" "	12.50	18.24	.498	.0232	16.19	21.58	.711	.0317
B and T	" "	12.10	18.21	.784	.0175	16.14	21.57	.528	.0173
Grand Ave	" "	12.22	18.23	.626	.0237	16.17	21.56	.736	.0304
True Value	" "	12.22	18.26	.819	.0309	-	-	-	-

None	Std. Dev.	5.18	2.56	.0681	.0207	4.43	3.00	.236	.0144
Branching	" "	5.16	2.51	.1665	.0140	4.48	3.01	.399	.0278
Time	" "	5.38	2.55	.1501	.0107	4.53	2.98	.183	.0249
B and T	" "	5.09	2.56	.3845	.0126	4.45	3.05	.202	.0114
Grand Ave.	" "	5.20	2.54	.1923	.0145	4.47	3.01	.255	.0196
True Value	" "	5.08	2.60			-	-	-	-

None	Prob. of Realizing	.128	-	.000368	-	.123	-	.000278	-
Branching	" " "	.131	-	.000389	-	.132	-	.000331	-
Time	" " "	.121	-	.000426	-	.120	-	.000384	-
B and T	" " "	.123	-	.000368	-	.124	-	.000215	-
Grand Ave.	" " "	.126	-	.000388	-	.125	-	.000302	-
True Value	" " "	.121	-	.000426	-	.121	-	.000426	-

APPENDIX III

	<u>Pages</u>
1. Some Features of GERT	III-1
2. How to Input Data	III-4
3. An Example	III-12
4. Variable List	III-17

III-1

The following text will explain in general some of the important features of the GERT program.

The program functions around the arrays or files NSET (the activity file) and QSET (the probability or time file). NSET is set up so that each node has associated (by means of pointers) with it, a file of activities emanating from the node. Every node can be considered a separate file in the range 2 to NOQ, except those nodes which do not appear in the network. File 1 which would be associated with node 1, if node 1 was allowed, is used to store all activities as they occur in the network and therefore will be called the working file. QSET contains, for the original data read in, the probability of an activity occurring. The values of QSET associated with the activities contained in the working file are the times at which the activity occurs.

The files are initially set up by Datan reading the activity information from cards and calling Filem to put the

activities in the activity file. The portion of NSET and QSET read in will be called the static file. Once the file is set up all activities are scheduled by Schat and put into the working file.

After an activity is taken from the working file (by Rmove according to lowest time first) and is realized, the following sequence (in the order given) of events happen:

- (1) the cost is accumulated,
- (2) the node counter is decremented,
- (3) activity cancellation(if specified),
- (4) node resets (if specified),
- (5) node modifications (if specified), and lastly
- (6) check to see if the node decremented has been realized. If the node has not been realized the program will remove another activity from the working file.

Once a node has been realized, the program (by Schat) immediately schedules all activities emanating from that node and puts them into the working file, along with the associated time.

The program will schedule and realize activities until either the network is realized or, it has found no more activities in the working file, in which case, an error occurs.

Figure III-1 gives an example of the network of Figure 3 during execution. The static file is contained in locations 15 through 26. The top half of Figure III-1 shows two activities in process in the working file. The bottom half shows the file at a later time.

FILE STATUS AT TIME	8.5732	ON RUN NUMBER	1	MFC(1) =	1	NQ(1) =	2	IBRNCH =	0	ITIME =	0
	QSET		NSET								
1	0.920445E 01	5	7	6	3	0	11	0	1	15	9999
2	0.100000E 01	2	3	7	1	0	0	0	0	7777	9999
3	0.100000F 01	3	4	2	2	0	1	4	2	7777	9999
4	0.100000E 01	4	5	3	2	C	4	2	1	5	9999
5	0.100000E 01	4	5	4	2	0	5	1	0	7777	4
6	0.100000E 01	5	7	5	2	0	0	5	1	7	9999
7	0.100000E 01	5	7	6	3	0	11	0	1	7777	6
8	0.100000E 01	10	11	5	7	0	9	5	1	7777	9999
9	0.100000E 01	7	8	3	2	0	0	2	2	7777	9999
10	0.100000E 01	11	8	3	2	0	10	2	2	7777	9999
11	0.500000E 00	8	4	1	1	0	2	0	0	13	9999
12	0.950000E 00	12	3	1	1	0	3	0	0	14	9999
13	0.100000E 01	8	12	1	1	0	0	0	0	7777	11
14	0.100000E 01	12	13	1	1	0	0	0	0	7777	12
15	0.109116E 02	5	7	5	2	0	0	5	1	7777	1
16	0.0	0	0	0	0	0	0	0	0	17	9999
17	0.0	0	0	0	0	C	0	0	0	18	16
18	0.0	0	0	0	0	C	0	0	0	19	17
19	0.0	0	0	0	0	0	0	0	0	20	18
20	0.0	0	0	0	0	0	0	0	0	21	19
21	0.0	0	0	0	0	C	0	0	0	22	20
22	0.0	0	0	0	0	0	0	0	0	23	21
23	0.0	0	0	0	0	0	0	0	0	24	22
24	0.0	0	0	0	0	0	0	0	0	25	23
25	0.0	0	0	0	0	0	0	0	0	26	24
26	0.0	0	0	0	0	0	0	0	0	9999	25

FILE STATUS AT TIME	10.9116	ON RUN NUMBER	1	MFC(1) =	15	NQ(1) =	1	IBRNCH =	0	ITIME =	0
	QSET		NSET								
1	0.0	0	0	0	0	0	0	0	0	16	9999
2	0.100000E 01	2	3	7	1	0	0	0	0	7777	9999
3	0.100000E 01	3	4	2	2	0	1	4	2	7777	9999
4	0.100000E 01	4	5	3	2	0	4	2	1	5	9999
5	0.100000E 01	4	5	4	2	0	5	1	0	7777	4
6	0.100000E 01	5	7	5	2	0	0	5	1	7	9999
7	0.100000E 01	5	7	6	3	C	11	0	1	7777	6
8	0.100000E 01	10	11	5	2	0	9	5	1	7777	9999
9	0.100000E 01	7	8	3	2	0	0	2	2	7777	9999
10	0.100000E 01	11	8	3	2	0	10	2	2	7777	9999
11	0.500000F 00	8	4	1	1	0	2	0	0	13	9999
12	0.950000E 00	12	3	1	1	0	3	0	0	14	9999
13	0.100000F 01	8	12	1	1	C	0	0	0	7777	11
14	0.100000E 01	12	13	1	1	0	0	0	0	7777	12
15	0.132436E 02	7	8	3	2	0	0	2	2	7777	9999
16	0.0	0	0	0	0	0	0	0	0	17	1
17	0.0	0	0	0	0	0	0	0	0	18	16
18	0.0	0	0	0	0	0	0	0	0	19	17
19	0.0	0	0	0	0	0	0	0	0	20	18
20	0.0	0	0	0	0	C	0	0	0	21	19
21	0.0	0	0	0	0	0	0	0	0	22	20
22	0.0	0	0	0	0	0	0	0	0	23	21
23	0.0	0	0	0	0	0	0	0	0	24	22
24	0.0	0	0	0	0	C	0	0	0	25	23
25	0.0	0	0	0	0	C	0	0	0	26	24
26	0.0	0	0	0	0	C	0	0	0	9999	25

Figure III-1

How To Input Data

<u>Data Card 1</u>		<u>Card Columns**</u>
Field 1	Analyst's name (6A2) (Name(6))	1-12
Field 2	Project number, if negative data card 11 is required (I4) (Nproj)	13-16
Field 3	Month (I2) (Mon)	17-18
Field 4	Day (I2) (Nday)	19-20
Field 5	Year (I4) (Nyr)	21-24
Field 6	Number of times network is to be simulated (I4) (Nruns)	25-28
Field 7	Number of parameter sets (I4) (Nprms)	29-32
Field 8	The largest number of activities which can be accomodated in the activity file. For most networks twice the number of nodes is sufficient. The maximum size of this is 500, although it can be expanded depending on the memory of the machine used. (I4) (Id)	33-36
Field 9	Integer random number seed (I10) (Jseed) any large odd number	37-46
Field 10	Starting value of cost histogram (F10.0) (Cstar)	N 47-56
Field 11	Cell width of cost histogram (F10.0) (Cinc)	N 57-66
<u>Data Card 2</u>		
Field 1	Largest node number used (I3) (Noq)	1-3
Field 2	Number of source nodes (I3) (Nsrc)	4-6
Field 3	Number of sink nodes (I3) (Npd)	7-9

**All numbers are right justified unless they can be placed anywhere in the field when punched with a decimal point in which case an 'N' appears before the column numbers.

Card
Columns**

Field 4	Number of sink nodes that must be realized before the network is realized (I3) (Nskst)	10-12
Field 5	Number of nodes on which statistics are to be collected, including all sink nodes (I3) (Nskst)	13-15
Field 6	Number of types of counts (Max of 4) (I3) (Ncts)	16-18
Field 7	A '1' if network modifications exists otherwise a blank or zero (I3) (Nnm)	19-21
Field 8	A '1' if node resets exist; '0' or blank otherwise (I3) (Nnr)	22-24
Field 9	A '1' if activity cancellation exists; '0' or blank otherwise (I3) (Loop)	25-27
Field 10	A '1' if antithetic time; '0' or blank otherwise (I3) (Itime)	28-30
Field 11	A '1' if antithetic branching; '0' or blank otherwise (I3) (Ibrnch)	31-33
Field 12	The number of times the program is to be restarted from run 1 using a different random number; if '0', '1', or blank, '1' is assumed (I3) (Lupdo)	34-36
Field 13	Option to print average time that node is first realized for every node in the network, the average time for the activity (difference of the averages of the two nodes the activity is between), percentage of time node is realized (number of times realized/number of runs); a '1' if print option wanted; '0' or blank if not (I3) (Nprttop)	37-39
Field 14	If logic node reset is used a '1' otherwise '0' or blank; if '1' then data card 3A is required (I3) (Logic) otherwise leave out	40-42

Δ Data Card 3Card
Columns**

Fields 1-25	The source node numbers followed by the sink node numbers followed by any other nodes on which statistics are to be collected. (25I3) (Nsrc(Nsrc), Nsink (Nsk))	1-3 4-6 . . . 72-75
-------------	--	------------------------------------

Data Card 3A

Needed only if a '1' is punched in Field 14 of Data

Card 2.

Field 1	The number of realizations of the logic reset node (I3) (Nrelp(1))	1-3
---------	---	-----

Δ Data Card 4

Fields 1-25	The number of realizations neces- sary to realize each node starting with node 2 and giving a value for each and every node sequentially until the maximum node number. If the realizations are zero it may be put in or left blank (25I3) (Nrelp (Noq))	1-3 4-6 72-75
-------------	---	---

Δ Data Card 5

Fields 1-25	The node type of each node from 2 to the end node with a value given for each. If the node is not used, zero or blank may be used. (25I3) (Ntype(Noq))	1-3 4-6 72-75
-------------	--	---------------------

Node Type 1 can be realized only
once per realization of the network.

Node Type 5 deterministic output -
is reset to original number of
releases.

Node Type 6 probabalistic output -
is reset to original number of
releases.

ΔWith at most 25 values per card for as many cards as needed.

		Card Columns**
<u>△△ Data Card 6</u>		
Field 1-8	The lower limit of the histograms for the sink nodes and for the nodes statistics are to be collected on, in the order in which they appear on Data Card 3. (8F10.0) (Xlow(Nsks))	N 1-10 N 11-20 . . . N 71-80

<u>△△ Data Card 7</u>		
Fields 1-8	The cell width of the histograms for the sink nodes and for the nodes statistics are to be collected in the order which they appear on Data Card 3, (8F10.0) (Width(Nsks))	N 1-10 N 11-20 . . . N 71-80

Data Card 8

The next (Nprms) cards with one parameter definition per card. (Param (Nprms,4))

Fields 1-4	(Depends on distribution) (4F10.0)	N 1-10 N 11-20 N 21-30 N 31-40
------------	------------------------------------	---

Distribution Type 1 - Constant

Field 1	The constant time
---------	-------------------

Fields 2-4	Not used
------------	----------

Distribution Type 2 - Normal

Field 1	Mean Value
---------	------------

Field 2	Minimum Value
---------	---------------

Field 3	Maximum Value
---------	---------------

Field 4	Standard Deviation
---------	--------------------

Distribution Type 3 - Uniform

Field 1	Not Used
---------	----------

Field 2	Minimum Value
---------	---------------

△△ With at most 8 values per card for as many cards as needed.

Field 3 Maximum Value

Field 4 Not Used

Distribution Type 4 - Erlang

Field 1 Mean time for the erlang variable divided
by the value given in Field 4.

Field 2 Minimum Value

Field 3 Maximum Value

Field 4 Number of exponential deviates to be included
in the sample obtained from the erlang
distribution.

Distribution Type 5 - Lognormal

Given μ = mean of lognormal

σ^2 = Variance of lognormal

Min Q = Smallest allowable value

Max Q = Largest allowable value

Field 1 $\ln \mu - 1/2 (\text{value of Field 4})^2$

Field 2 $\ln (\text{min } Q)$

Field 3 $\ln (\text{max } Q)$

Field 4 $\sqrt{\ln (\sigma^2/\mu^2 + 1)}$

Distribution Type 6 - Poisson

Field 1 Mean minus minimum value

Field 2 Minimum Value

Field 3 Maximum Value

Field 4 Not Used

The interpretation of the mean for the poisson distribution should be the mean number of time units per time period.

If the random time generated is greater than the maximum allowable or less than the minimum allowable, the time will be set to whichever limit was exceeded.

Data Card 9Card
Columns**

This card inputs the activity and probability files.

Field 1	Probability of activity occurring (F8.0) (Atrib)	N 1-8
Field 2	Start node (I3) (Jtrib(1))	9-11
Field 3	End node (I3) (Jtrib(2))	12-14
Field 4	Parameter number (I3) (Jtrib(3))	15-17
Field 5	Distribution type (I3) (Jtrib(4))	18-20
Field 6	Count type (I3) (Jtrib(5))	21-23
Field 7	Activity number (I3) (Jtrib(6))	24-26
Field 8	Variable cost for activity (I3) if zero can be left blank (Jtrib(7))	27-29
Field 9	Fixed cost for activity (I3) if zero can be left blank (Jtrib(8))	30-32

The last card of this type has a zero or blank for
Field 2.

*Data Card 10

Only required if network modifications exist; a '1' in
Field 7 of Data Card 2.

Field 1	Activity number (I3)	1-3
Field 2	Number of node to be modified (I3)	4-6
Field 3	Number of node inserted (I3)	7-9
Fields 4-25	Repetition of Fields 2 and 3 (Naba(1,K))	72-75

The last data card of this type must have a blank or
zero in Field 1.

*For Data Cards 10, 10A and 10B continuation may exist. The
continuation card must have the same activity number and must
follow immediately the card being continued. The number of
continuation cards allowed only depends on the memory allowance.

*Data Card 10AData
Columns**

Only required if using node resets. A '1' punched in Field 8 of Data Card 2.

Field 1	Activity number (I3)	1-3
Field 2	Node number to be changed (I3)	4-6
Field 3	Number of node which has the original number of releases that is required for the node in Field 2 (I3)	7-9
Field 4-25	Repeat Field 2 and Field 3 (Naba(2,K))	10-12
		73-75

The last card of this type must have a zero or blank in Field 1.

*Data Card 10B

Only required if activity cancellation is required. A '1' punched in Field 9 of Data Card 2.

Field 1	Activity number (I3)	1-3
Field 2	The node number from which activities in process emanate are destroyed (I3)	4-6
Field 3-25	Repetition of Field 2 (Naba(3,K))	7-9
		73-75

The last card of this type must have a zero or blank in Field 1.

Data Card 11

Only used if project number (Data Card 1, Field 2) is negative.

Field 1	The run number for which tracing should begin (I3) Naba(1, Imn)	1-3
---------	---	-----

Field 2	The run number for which tracing should end (I3) (Naba(1, Imn-1))	4-6
---------	--	-----

Multiple networks may be simulated by stacking the data cards for each network one after the other with no intervening cards.

(Example of Input and Output)

To illustrate the preparation of data the following example will be used. It should be noted that the network is that given in Figure 3. The costs used can represent mental and physical exertion. The input data is given in Figure III-3. The output is given in Figure III-4 and Figure III-4A.

Figure III-4 shows what was originally read in and gives a small portion of tracing. Figure III-4A shows the answers for time and cost. The bottom of this figure illustrates what is printed, if the print option is specified (Data Card 2, Field 13). The 'Time/End Node' column gives the average time at which the end node shown is first realized. The 'Time/Activity' takes the difference of the 'Time/End Node' for the start and end nodes of the activity. The 'Percent' column is calculated as the number of times the end node shown is first realized divided by the total number of realizations of the network.

PRECEDING PAGE ~~MISSING~~ NOT FILMED

III-13

7.0	A	CLARK	53	6	11	70	50	7	26	5271	0.0	20.0	GF010
13	1	1	1	0	1	0	0	0	0	1	0		GF020
2	13												GF030
0	1	1	2	0	2	1	0	2	1	1	1		GF040
1	0	0	0	0	0	0	0	0	0	6	1		GF050
0.0													
50.0													
0.0													GF080
1.0		0.0			2.0			0.3					GF090
3.0		0.0			6.0			1.0					GF100
0.0		0.0			10.0			2.0					GF110
2.0		0.0			3.5			0.0					GF120
		0.0			1.0								GF130
1.0			2	3	7	1	0	0	0				
1.0			0	4	2	0	0	1	0				GF150
1.0			4	0	0	0	0	4	0				GF160
1.0			4	0	0	0	0	0	0				GF170
1.0			5	7	5	0	0	0	5	1			GF180
1.0			5	7	6	0	0	1	0	1			GF190
1.0		10	11	5	2	0	0	0	5	1			GF200
1.0		11	0	0	0	0	0	0	2	2			GF210
1.0		11	0	0	0	0	0	0	2	2			GF220
0.0		0	4	1	0	0	0	0	0	0			GF230
0.95		12	3	1	0	0	0	0	0	0			GF240
0.5		8	12	1	0	0	0	0	0	0			GF250
0.05		12	13	1	0	0	0	0	0	0			GF260
		0											GF270
2	0	10											GF280
3	10	0											GF290
0													GF300
1	1												GF310

NOT REPRODUCIBLE

Figure III-3. Input Data

GERT SIMULATION PROJECT -33 BY R. A. CLARK
DATE 6/ 1/ 1970

NETWORK DESCRIPTION

START NODE	END NODE	PARAMETER NUMBER	DISTRIBUTION TYPE	COUNT TYPE	ACTIVITY NUMBER	VAR. COST	FIX. COST	PROBABILITY
2	3	7	1	0	0	0	0	1.0000
3	4	2	2	0	1	4	2	1.0000
4	5	3	2	0	4	2	1	1.0000
4	5	4	2	0	5	1	0	1.0000
5	7	5	2	0	0	5	1	1.0000
5	7	6	3	0	11	0	1	1.0000
7	8	3	2	0	0	2	2	1.0000
8	4	1	1	0	2	0	0	0.5000
8	12	1	1	0	0	0	0	0.5000
10	11	5	2	0	9	5	1	1.0000
11	8	3	2	0	10	2	2	1.0000
12	3	1	1	0	3	0	0	0.9500
12	13	1	1	0	0	0	0	0.0500

ACTIVITY PARAMETERS

PARAMETER NUMBER	PARAMETERS			
	1	2	3	4
1	0.0	0.0	0.0	0.0
2	1.0000	0.0	2.0000	0.3000
3	3.0000	0.0	6.0000	1.0000
4	5.0000	0.0	10.0000	2.0000
5	2.0000	0.0	3.5000	0.5000
6	0.0	0.0	1.0000	0.0
7	1.0000	0.0	0.0	0.0

NETWORK MODIFICATIONS

IVITY NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE NODE FILE

2 5 10 0
3 .10 5 0

HIGHEST NODE NUMBER IS 13
NUMBER OF SOURCE NODES IS 1
NUMBER OF SINK NODES IS 1
NUMBER OF NODES TO REALIZE THE NETWORK IS 1
STATISTICS COLLECTED ON 1 NODES
NUMBER OF PARAMETER SETS IS 7
INITIAL RANDOM NUMBER IS 5271
ANTITHETIC TIME INDICATOR IS 0
ANTITHETIC BRANCH INDICATOR IS 0

SOURCE NODE NUMBERS

2

SINK NODE NUMBERS

13

NODE	NUMBER RELEASES	NODE TYPE
1	0	0
2	0	1
3	1	5
4	1	5
5	2	5
6	0	0
7	2	5
8	1	6
9	0	0
10	2	5
11	1	5
12	1	6
13	1	1

AT TIME 1.00 ACTIVITY ON NODE 3 WITH ATTRIBUTES 7 1 0 0 0 0 WAS REALIZED ACCUMULATED COST = 0.0
 AT TIME 2.25 ACTIVITY ON NODE 4 WITH ATTRIBUTES 2 2 0 1 4 2 WAS REALIZED ACCUMULATED COST = 7.01
 AT TIME 4.84 ACTIVITY ON NODE 5 WITH ATTRIBUTES 3 2 0 4 2 1 WAS REALIZED ACCUMULATED COST = 13.17
 AT TIME 8.57 ACTIVITY ON NODE 5 WITH ATTRIBUTES 4 2 0 5 1 0 WAS REALIZED ACCUMULATED COST = 19.50
 AT TIME 9.28 ACTIVITY ON NODE 7 WITH ATTRIBUTES 6 3 0 11 0 1 WAS REALIZED ACCUMULATED COST = 20.50
 AT TIME 10.91 ACTIVITY ON NODE 7 WITH ATTRIBUTES 5 2 0 0 5 1 WAS REALIZED ACCUMULATED COST = 33.19
 AT TIME 13.24 ACTIVITY ON NODE 8 WITH ATTRIBUTES 3 2 0 0 2 2 WAS REALIZED ACCUMULATED COST = 39.85
 AT TIME 13.24 ACTIVITY ON NODE 12 WITH ATTRIBUTES 1 1 0 0 0 0 WAS REALIZED ACCUMULATED COST = 39.85
 AT TIME 13.24 ACTIVITY ON NODE 3 WITH ATTRIBUTES 1 1 0 3 0 0 WAS REALIZED ACCUMULATED COST = 39.85
 NODE 10 WAS CHANGED TO NODE 5 AND SET TO 2
 AT TIME 13.98 ACTIVITY ON NODE 4 WITH ATTRIBUTES 2 2 0 1 4 2 WAS REALIZED ACCUMULATED COST = 44.78
 AT TIME 17.51 ACTIVITY ON NODE 5 WITH ATTRIBUTES 3 2 0 4 2 1 WAS REALIZED ACCUMULATED COST = 52.85
 AT TIME 22.10 ACTIVITY ON NODE 5 WITH ATTRIBUTES 4 2 0 5 1 0 WAS REALIZED ACCUMULATED COST = 60.98
 AT TIME 22.88 ACTIVITY ON NODE 7 WITH ATTRIBUTES 6 3 0 11 0 1 WAS REALIZED ACCUMULATED COST = 61.98

GERT SIMULATION PROJECT -33 BY E. A. CLARK
DATE 6/ 1/ 1970

FINAL RESULTS FOR 50 SIMULATIONS

NODE	PROB./COUNT	MEAN	STD.DEV.	MIN.	MAX.
13	1.0000	400.4487	299.7185	10.3260	1023.9724

HISTOGRAMS

NODE	LOWER LIMIT	CELL WIDTH	FREQUENCIES										
13	0.0	50.00	0	3	5	5	2	5	3	3	4	2	3
			1	0	1	2	2	1	3	1	1	0	3
			0	0	0	0	0	0	0	0	0	0	

HISTOGRAMS

NODE	LOWER LIMIT	CELL WIDTH	FREQUENCIES										
COST	0.0	20.00	0	0	1	0	1	0	0	1	0	1	1
			1	0	1	0	0	0	1	1	3	0	0
			0	1	0	0	1	0	0	0	0	36	

AVE. COST PER REALIZATION 1280.54
STANDARD DEVIATION 959.73
MIN. COST FOR REALIZATION 35.74
MAX. COST FOR REALIZATION 3334.73

NODE	START	END	WITH ATTRIBUTES					TIME/END NODE	TIME/ACTIVITY	PERCENT
2	3	7	1	0	0	0	0	1.00	1.00	1.00
3	4	2	2	0	1	4	2	2.03	1.03	1.00
4	5	3	2	0	4	2	1	7.41	5.38	1.00
4	5	4	2	0	5	1	0	7.41	5.38	1.00
5	7	5	2	0	0	5	1	9.44	2.03	1.00
5	7	6	3	0	11	0	1	9.44	2.03	1.00
7	9	3	2	0	0	2	2	12.46	3.02	1.00
8	4	1	1	0	2	0	0	2.03	-10.43	1.00
8	12	1	1	0	0	0	0	26.85	14.39	1.00
10	11	5	2	0	0	5	1	29.04	1.90	0.96
11	8	3	2	0	10	2	2	12.46	-16.58	1.00
12	3	1	1	0	3	0	0	1.00	-25.85	1.00
12	13	1	1	0	0	0	0	400.45	373.60	1.00

Figure III-4A

A list of variables not explained in the input data description.

ACTIME (NOQ)	Stores the time a node has been realized.
ATIME (MAXA)	Array used to store the values of antithetic times.
ATRIB	Used to transfer values to and from Qset.
BRNCH (MAXA)	Used to store antithetic branching probabilities.
CMAX	Stores maximum cost.
CMIN	Stores minimum cost.
COSSQR	Stores the $\sum_i (\text{cost}_i)^2$.
COST	The cost for realization.
CSAVE (NOQ)	Counts the number of times the node has first been realized.
IM	Number of entries for an activity, excluding the two pointers, in NSET.
IMM	Specifies number of antithetic variables stored.
IMN	Specifies size of array NABA
ISEED	Random number generator seed.
JCELS (NHIST,MXC)	Array used to store histogram information.
KLE	Last entry in file indicator (9999)
KOF	Last entry of array NSET indicator (8888)
KOL	Beginning of node file indicator (7777)
KOUNT (NCTS)	Used to store count statistics.
KOWT	Used as error counter.
MAXA	Maximum number of locations for antithetic storage.
MAXNS	Maximum number of locations of NSET used = ID*MXX
MFA	First available location in NSET for next activity
MFE (NOQ)	First entry in a file.

MLE(NOQ)	Last entry in a file.
MX	= IM+1
MXC	Number of cells in histogram.
MXX	= IM+2
NCLCT	Number of sets of statistics that can be collected in COLCT.
NCRDR	Number used for card reader.
NFTBU(NOQ)	Used to store the next node to be used.
NHIST	The number of nodes that have histograms = NSKS
NPO (3, activity number)	Stores the location of modifications in NABA by activity number.
NPRNT	Number of the printer
NQ(NOQ)	Current number of entries in any file.
NREL(NOQ)	Number of releases to go for each node.
NRUN	Counter incremented after each realization until it = NRUNS
NSET(MAXNS)	Stores all activities and the activity working file.
NSKSR	= NSKST
NTYPE(NOQ)	Stores the node type for each node.
QSET(ID)	Stores the probability or time for each activity.
SUMA(NSKS,5)	Stores the statistics collected
TIBR	Used for antithetic time and branching.
TNOW	The time now of a realization.
TOTIM	Total Time.
TSAVE(NOQ)	Saves the sum of the times the node is first realized.