UTILITY OF DATA BASE MANAGEMENT TO ANALYZE THE OUTPUT FROM COMPLEX SIMULATIONS

Pamela E. Joseph
Analytic Services, Inc., Falls Church, Va.
Stephen D. Roberts, Ph.D.
Purdue University, Indiana University
Indianapolis, Indiana

Abstract

The analysis of the outputs generated by high level simulation languages from complex systems models is often hindered by both the statistics collection capability of the simulation language and the foresight employed by the modeler in the model construction. Augmenting a simulation language (INS) with a data base management system (GPLAN) is one method for alleviating the problems of data (event) inaccessibility and specific inquiry. The development of the data structure for use in filing the information generated during a simulation, the description of this structure to GPLAN via the data description language, and the use of the structure through the data manipulation language commands for storing and retrieving the information are described. The expanded capabilities for investigating the model become evident upon the description of the reports and the ease with which the reports may be expanded due to the modular design of the data base program. Also the modeler's investigation of some portion of the system or the entire system is enhanced since through the data base any piece of information that has been stored can be easily accessed. The result indicates that a data base management system can be used successfully to augment the use of simulation in the investigation of a complex system. Considerations regarding the computer execution time and storage are also presented.

INTRODUCTION

The development of high level simulation languages has provided a convenient capability to model complex systems that give rise to the generation of large and elaborate amounts of output. When there is significant output, the statistics collection capabilities of the simulation language may not provide the modeler with the information needed. Statistics collected by a simulation language often provide information that is too general, and if more specific information is desired a dump of state changes from the modeled system must be analyzed. Other problems may appear in the analysis of the system model depending on the foresight of the modeler in constructing the model to collect appropriate information.

The utilization of a data base management system (dbms) with the simulation language can help overcome the problems mentioned above. The dbms would allow the modeler to store the detailed output from the simulation in a data base. Once stored, information may be accessed by the modeler as frequently as desired without the necessity of rerunning the simulation model or reconstructing the statistics collection mechanisms. The modeler may design special simulation reports via dbms commands to accomodate particular needs. With each access of the data base different reports may be requested, thus allowing a thorough investigation of a model in a manner not possible with a fixed set of standard reports provided by the simulation language. It also becomes easier to expand the output of the simulation to include other reports difficult to implement with the simulation language package because with the dbms the simulation package does not need to be modified if the output of the simulation provides a complete description of the modeled system. The option is also made available for reports which might require more than one pass through the data base.

Based on the application of these ideas, the utility of a dbms to analyze the output from complex network simulations will be discussed. The simulation language used for this application is INS, a new network simulation language that provides for the convenient high level modeling of complex resource constrained queuing systems. GPLAN, a network based dbms implementation, is used as the data base system. The discussion includes the methodology followed in designing the data base and the application program, followed by a comparison of computer execution time and memory requirements, and concluding with a discussion on the information storage requirements of the data base for larger simulations.

INS

The Integrated Network Similation language, INS, is a FORTRAN (GASP IV) based simulation language that was developed to help analyze complex social systems, particularly health care systems, or other systems that display similar characteristics (1,2,3). INS allows the modeler

to integrate the networks of two active entities. One entity represents demands for services (transactions) while the second represents the resources that respond to the services required by the transactions. No programming by the modeler is required in order to use INS and its convenient input/output makes the language simple to learn and use.

INS employs a syntax consisting of parameterized nodes and branches which can be graphically presented as a network. The powerful nodes have rich semantic utility and can represent a broad set of problems. The basic nodes in the INS network provide for:

a) creating transactions and scheduling their arrival - at a source node;
b) representing the departure of transactions by removing them from the network - at a sink node;
c) routing of transactions through the network - at a route node;
d) assigning transaction attributes - at an assignment node;
e) collecting statistics on transit times of transactions between two points in the network - at a mark node;
f) representing an activity which may consume resources and synchronize transaction processing - at an activity node;
g) delays at an activity node due to resource unavailability, process synchronization, or queue discipline - at a queue node;
h) representing the decision process used by a resource in selecting a waiting transaction on which to perform an activity -- uses a selector node; and
i) branches transactions between nodes, expressing their precedence and specifying a selection rule while at the same time providing a facility to endogenously create transactions, specify synchronization for transactions, and transfer resources within a transaction set - using a branch.

INS features capabilities similar to other network-oriented languages such as GPSS (4) and Q-GERTS (5), but uniquely provides for the multi-activity, multi-resource characteristics of queuing systems. This can mean:

a) activities have multiple resource requirements;
b) substitution among resources can exist;
c) resources may perform any number of activities and can sometimes choose among these activities according to complex criteria;
d) some activities may preempt others for resources; and
e) resources arrive and depart in different manners.

Thus INS models, while easily developed, can present a complex and detailed system. Such a high level capability presents a broad need of output anlysis, from macro measures of system performance to the micro detail of the movement of particular transactions.

Output available from INS (version 2) can be categorized in three sections. The first is a display of input records and an echo of the INS model organized about the network elements.

The second output is optional but provides a time ordered listing (state trace) of all the changes of state (subevents) that occurred during the simulation. These data plus the network structure information completely describe the simulation run. The state trace can obviously be a lengthy document and tedious to analyze.

The third output comprises the simulation summary report and provides an aggregation of output. With the exception of one report, all the elements of this report are automatically produced. The summary report consists of:

a) overall simulation information dealing with the simulation runs and statistics collection;
b) time statistics for queue, activity, mark, and sink nodes for each transaction type and/or all transaction types;
c) queue length statistics;
d) aggregated queue length statistics on groupings of queues which form a logical queue;
e) resource utilization information;
f) (optionally) resource utilization by individual activity; and
g) average number of transactions in the systems according to their origin.

Statistics can be collected over a single run or over several runs and the state trace can be similarly specified.

Unfortunately, a considerable gap exists between the summary data and the state trace. The modeler wishing deeper insight than summary information must either remodel the network or request the state trace and search the listings for the desired information--a process which may involve literally thousands of state changes in a complex model. Hence the opportunity for examining closer the simulation output is greatly hindered.

Because of these considerations the application of a data base system is seen to be appropriate. The data base would consist of the detailed simulation output, but through the capabilities of a dbms the modeler would have greater freedom in choosing and tailoring output, independent of the simulation model.

GPLAN

The data base system chosen for accomplishing the task of handling the simulation output was GPLAN (Generalized PLANning System), a data base management system developed at Purdue University (6,7). GPLAN is a FORTRAN implementation of a network based dbms, following the guidelines established by the CODASYL Data Base Task Group (DBTG) in the April, 1971 report (8). Because of the host language and its availability, GPLAN was considered an ap-

propriate choice for use with INS to develop the data base approach to analyzing simulations. The network approach to dbms was further considered to be very applicable to the numeric data base output from a simulation.

GPLAN makes it possible for the modeler to design a data base that will meet specific requirements and represent accurately the relationships inherent in the data. The modeler translates a pictorial representation of the data base into language understandable by the GPLAN program, the Data Definition Language (DDL). The DDL allows the user to describe the data base structure in terms of sets, records, and data items. Data items are the smallest units of information in the data base, and they are usually represented as fields within records, where records are collections of none, one, or more data items. Sets are named collections of records, form the building blocks for the data structure by linking together records, and are ordered so that information entered into the data base is entered in a particular manner. For this application all the sets are ordered such that the first information placed in a record will be the first information accessed from that record, although there are other types of set ordering available in GPLAN. Set membership identifies one record as an owner and at least one record declared as a member.

Using these building blocks and the definitions either hierarchical (tree) structures or network structures can be designed. Tree structures are such that any member record has a single owner, although each owner record may have more than one member defined. Network structures, on the other hand, allow member records to have more than one owner. The network capability was one of the appealing features of GPLAN, since the INS data was naturally portrayed in a network form.

Once the data structure has been defined through the DDL, the data base is loaded. The loading is accomplished using the Data Manipulation Language (DML). Once the data has been loaded, the data base is ready for use. The DML is used to locate the appropriate record and then copy the information from the record for the modeler's use. Hence, the DML amounts to a set of FORTRAN subroutine calls that search through the data base for the correct record and inserts or retrieves the information located there. Other DML commands delete information and create new sets, records, or data items, among other commands (6,7).

DATA BASE DESIGN

The first step was to design the data base structure premised on what data were needed, how the data were related, and what supporting data (system description data) were needed. This was accomplished by examining the simulation state trace from which all the information needed to calculate all the statistical reports can be found.

To link the possible records within the data base design, a number of criteria were identified. These included:

a) minimizing the duplication among records;
b) ease of storage and retrieval; and
c) movement among records for information minimized DML commands.

In addition to the ability to conveniently and efficiently store simulation state change information, the parameters describing the simulation model structure were also stored. Possessing the model structure, the data base could process statistical reports independent of INS. Hence the data base would contain the experienced interaction between transactions and resources during the simulation.

The data structure that evolved is shown in Figure 1, and depicts an expanded view of the state trace. The blocks represent records, the names within the records are data items, and the records are linked by sets. The seven records on the left half of Figure 1 represent the INS state trace information while those on the right half contain system information on the INS model structure. The record SYSTEM is provided by GPLAN and allows the modeler entrance into GPLAN. SYSTEM is the starting point from which all the records are defined and the set linkages declared.

The record ACT contains activity information, specifically the activity number (ANUM) and the activity type (ATYP), which is a modeler specified classification indicating how that activity should be processed during the simulation. The record TRAN is a key record for the data structure. The data items here are EVNT, which describes the particular state change that has occurred; TNUM, which is the transaction number, one of the two ways in which transactions are identified; and TYPE, the transaction type, the other transaction identifier. The record TRAN is important because of EVNT, which is accessed each time a new state change (subevent) is to be processed. The determination of EVNT signals which of the other state trace related records has to be accessed also in order to collect all the information that describes the subevent that occurred. Since almost every subevent (with the exception of three of fourteen subevents) involves transactions, EVNT was placed in TRAN. The next record is RESC, which contains resource information relating to the subevents also involving transactions. The data items are the number of resources utilized in the subevent (RNBR), the resource number of each one utilized (RNUM), and the resource type for each (RTYP). Like transactions, resources are identified in two ways so the modeler can identify at each activity if a specific resource is needed (by specifying its number) or if any resource of a given type will be satisfactory (by specifying the type). The record RES2 identifies by number (RNMR) and type (RTPE) the resource involved in the current subevent that does not involve a transaction. These subevents describe the arrival of a resource, a resource entering the idle state, and a resource departing the idle state. The record CLOK identifies the TIME a subevent occurred. Besides TRAN, CLOK is the only other record that will always be accessed. Information about queues is contained in the record QUES. Each queue is identified by a queue number (QNUM), and the number of transactions in the queue at that time is

given by QPOS. The last data item in the record, AQNO, is used if the queue belongs to an aggregated queue. The last state trace based record is TRN2, which contains information about those subevents in which only transactions are involved, i.e., when transactions arrive and leave the system and are created or destroyed. The data items indicate the node number (TNOD) involved and give a count of the number of primary transaction (NPRI) and the number of endogenously derived transactions (NDRV) in the system at the time.

The last three records contain system information. The records and their data items are translated in Figure 2. The record PRAM contains parameter information which indicates to the access program how large statistics collection arrays need to be, how many simulation runs will be made and the total time the simulation spans. The record TLST provides a listing, if any, of the transaction type reference lists. These lists are used in determining how statistics should be collected at various nodes. TLST is used in conjunction with the record LIST, which contains information on activities, queues, and sink nodes at which statistics are wanted. The information also indicates how the statistics are to be collected and which transaction type reference list, if any, is pertinent to the collection.

USING THE DATA BASE

Once the data structure has been designed, it can then be described with the DDL and run through the DDLANL, which analyzes the description and establishes the structure in the computer. Space is allocated and pointers are initialized. Pointers are used to move through the data base, to keep track of what set, record, and data items have been most recently accessed.

After the design of the data base and its establishment, the next task is to load it with the desired information. For this purpose a load program was developed which utilized DML (data manipulation language) commands to move from SYSTEM to the appropriate record and then to place the information in the correct data items. Once a record has been loaded, a data base key is returned. The key gives the location at which the information was stored within the data base. If a key has been returned, then the loading is correct and the next piece of information may be loaded.

Once the data base has been loaded, it is ready for use in applications as desired by the modeler. The access program as it was developed for this application gives the modeler a choice of reports. Two of the reports require the modeler to specify the boundaries within which the reports should be constructed. At this time the report options available include all those reports available from INS plus some additional reports. The duplication of INS reports was to test the feasibility of this approach. The access program was developed in modular form so that other report options could be added as they were developed and so be incorporated into the entire program with the minimum of modification to the already existing modules.

The modeler, in using the access program, first indicates if any of the standard INS reports are wanted. Figure 3 contains a set of example user inputs while Figure 4 echos the interpreted inputs. Refering to the first line of Figure 3, he can request respectively the state trace, the discrete (time delay) statistics report, the average queue length report, the average aggregated queue length report, the average resource utilization report, the resource utilization by individual activity report, and the average number of transactions in the system report. The last yes/no on the first line corresponds to the modeler requesting a report for which he has to make specifications. If the answer is negative, then the modeler input is finished and the access program begins processing the reports requested. If the answer is affirmative, the modeler then specifies if he wants mark statistics (MRK on line 2) and/or a selective state trace (STR either on line 2 or some even line number multiple after). Mark statistics give values for average transit times between two points within the system, and if these statistics are wanted, they are requested next. When the modeler specifies mark statistics, the request is followed by the starting point and the ending point of the collection and a description of how the statistics are to be collected. For example, in Figure 3 mark statistics are collected from the time transactions arrive at node 1 until the activity at node 11 had been completed. The statistics are to be collected by transaction type (T)according to the types listed in transaction type reference list number 3. All the mark statistics wanted by the modeler are listed before specifying the type of selective trace desired.

The modeler has a choice of three selective trace options. One option traces a specified transaction type from the time it arrives in the system to the time it departs the system. A second option traces a specified resource over the entire simulation from the time it arrives until the simulation has been completed. The third option traces a specified resource between modeler specified starting and stopping times. If the modeler wants the selective trace, it is generated on a second pass through the data base. All the other reports, including the state trace if it has been requested, are generated on one pass through the data base.

In Figure 3, a trace of resource number (RNM) 5 over the entire simulation is specified. The output from that request is given as Figure 5. Hence, the selective trace provides the modeler a capability for expanding the investigation of the modeled system that is not available with INS unless provision is made for a large temporary storage space so the desired subevents may be saved until the end of the simulation.

UTILITY OF THE DATA BASE SYSTEM

The GPLAN application allows the modeler to investigate a model in a more disaggaregated manner by appropriately selecting reports than it does with INS. Because the modeler has access to the data, he can control the output. Thus, while the statistical reports aggregate the data, the modeler can at the same time have access to the pieces that comprise the aggregated report, to form smaller aggregated reports and through the selective trace to obtain specific transaction or resource information to generate additional insight. With this data availability, the modeler is better equipped to investigate and comprehend the system being modeled.

The use of the mark statistics is especially flexible. With the GPLAN application the modeler specifies the portion of the network over which the mark statistics are to be collected. The segment of interest may change from simulation run to simulation run without invoking INS to regenerate the data each time. In INS in order to redefine the mark statistics collection, a new mark node has to be defined and the simulation model rerun. The model modification is not necessary in the data base system since the modeler need only specify the beginning and ending points for the statistics collection, and these points may vary from run to run.

Thus, at this time, the greatest utility of the GPLAN application lies in the modeler's ability to access the information in the data base selectively to augment or verify the aggregated statistics reports requested and to selectively segment the network into managable pieces to allow investigation of subsets of the entire network. Both of these capabilities enable the modeler to expand his investigative capacities by allowing greater access to the data and greater control over the investigation of smaller segments of the network.

EXECUTION

The data base system allows the simulation output to be analyzed any number of times without resimulating. The access program takes less time to execute and uses less computer central memory than INS for the models tested. The descrepancy is not surprising when it is remembered that INS compiles and executes the simulation in its time and computer space besides generating the reports. The comparisons in time and space were made for access program requests that yielded the same reports as INS and showed for the examples used to test the access program that the access program had memory space savings of more than 10% ($67{,}200_8$ words for INS vs. $60{,}500_8$ words for the access program), saved better than 9 seconds in central processor time to execute the entire program (33.2 seconds for INS vs. 23.5 seconds for GPLAN), and saved more than 2 seconds in the time needed to load and execute the program (7.33 seconds for INS vs. 5.31 seconds for GPLAN). The time requirements varied, as one might expect, with the complexity of the system being modeled.

INFORMATION STORAGE

When the data base becomes extremely large, the time advantage of the data base program will be lost because the data base will probably have to be stored on an auxiliary storage device such as a magnetic tape. The data base becomes extremely large when more complex systems are modeled or when systems are modeled for longer periods of simulated time. Even though the output may not be large after running a long simulation, insertion of the information into the data base will greatly expand the size of the overall data base because of the number of pointers required by GPLAN to keep track of the records and sets defined.

Thus when auxiliary storage is needed, the attractiveness in using the GPLAN application will be diminished due to the longer program execution times. It will be necessary for the potential user to decide if the longer execution time is worthwhile in order to gain the greater access to the data for a more comprehensive investigation of the system.

CONCLUSION

It was demonstrated that augmenting INS with GPLAN was feasible and appropriate. The data base allows for an extension of the simulation by allowing the modeler greater access to the data generated by a simulation, giving greater flexibility in choosing the reports to be generated, and giving a consistency in reports as they are compared to those of other users working with the same model.

BIBLIOGRAPHY

1. Roberts, S.D., Fox, M., Sadlowski, T.E., and Kronman, B.K., "INS User's Manual for Version 2," Regenstrief Institute for Health Care, Indianapolis, Indiana, 1977.

2. Roberts, S.D, and Sadlowski, T.E., "INS: Integrated Network Simulator," Proceedings of the 1975 Winter Computer Simulation Conference, December, 1975.

3. Roberts, S.D., and Sadlowski, T.E., "INS: A Simulation Language for Analyzing Operational Issues in Ambulatory Care," ORSA/TIMS Conference, November 1976.

4. Schriber, T.S., Simulation Using GPSS, John Wiley and Sons, New York, 1974.

5. Pritsker, A.A.B., Modeling and Analysis Using Q-GERT Networks, Halstead Press, New York and Pritsker & Associates, Inc., Lafayette, Indiana, 1977.

6. Whinston, A. and Haseman, W., Introduction to Data Management, Richard D. Irwin, Inc., Homewood, Ill., 1977.

7. Bonczek, R., Cash, J., Haseman, W., Halsapple, C., and Whinston, A., "Generalized Planning System/Data Management

System (GPLAN/DMS) Users Manual," Krannert Graduate School, Purdue University, West Lafayette, Indiana, November, 1975.

8. CODASYL Data Base Task Group Report, April, 1971.

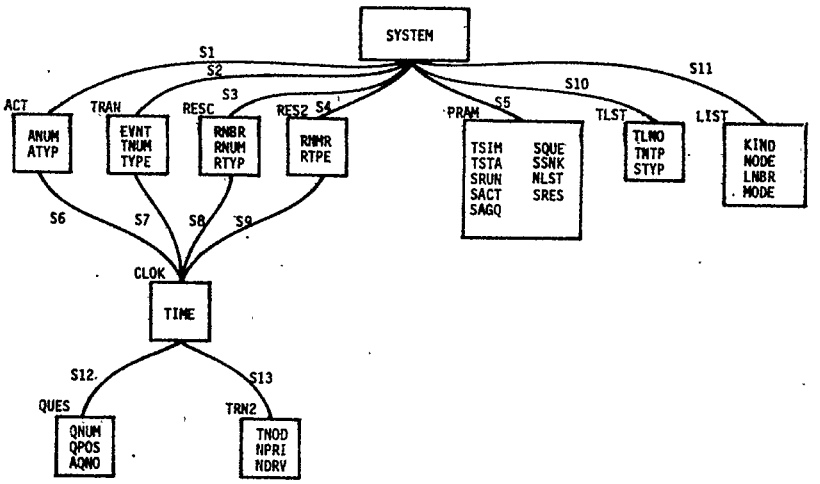

Figure 1: Data Base Logical Structure

SET S5	Parameters
OWNER - SYSTEM MEMBER - PRAM	Parameters
ITEM - TSIM	Time simulation began
ITEM - TSTA	Time statistics collection began
ITEM - SRUN	Number statistical collection runs made
ITEM - SACT	Total number activity nodes for discrete stats
ITEM - SAGQ	Total number queue nodes involved in aggregated queue statistics
ITEM - SQUE	Total number queue nodes for discrete stats
ITEM - SSNK	Total number sink nodes for discrete statistics
ITEM - NLST	Number of transaction type reference lists specified
ITEM - SRES	Number of resources specified for system
SET S10	Transaction type reference lists
OWNER - SYSTEM MEMBER - TLST	Transaction type reference lists
ITEM - TLNO	Transaction type reference list number
ITEM - TNTP	Number of transaction types in list
ITEM - STYP	Listing of transaction types
SET S11	Selected Node Information
OWNER - SYSTEM MEMBER - LIST	Node information on Activities, Queues, and Sinks
ITEM - KIND	Node type - QUE,ACT,SNK
ITEM - NODE	Node number
ITEM - LNBR	Transaction type reference list number
ITEM - MODE	Discrete statistics collection mode

Figure 2: Record Descriptions for System Information Records

```
NO ,YES,NO ,NO ,NO ,NO ,YES,YES
MRK
ARV   1 END  11 T     3
RNM  5
```

Figure 3: User Inputs

```
************************
THE SIMULATION REQUESTS
************************

** GENERAL STATISTICAL REPORTS **

   DISCRETE STATISTICS
   AVERAGE NUMBER OF TRANSACTIONS IN THE SYSTEM


** SPECIFIC STATISTICAL REPORTS **

                                STAT TRAN
          BEGIN NODE  END  NODE COLC LIST
          STATE NBR  STATE NBR  MODE NBR

   MARK999  ARV    1  END    11   T      3


** STATE TRACE LISTING **

   SELECTIVE TRACE FOR RESOURCE NUMBER    5 OVER ENTIRE SIMULATION
```

Figure 4: Echo Check

```
**********************************************************************
DYNAMIC SYSTEM SELECTIVE STATE TRACE - SIMULATION RUN 1
**********************************************************************

          TRACE THRU SYSTEM RESOURCE NUMBER    5

                    ACT ACT QUE NBR IN TRN TRN
     TIME STATE NBR NBR TYP NBR  QUEUE NBR TYP
      .00 ARV           IDL
      .00 REQ           IDL
     3.29 RDQ
     3.29 BGN   17  RI                  1   1
     7.88 END   17  RI                  1   1
     7.88 BGN    2  RI                  1   1
    12.88 END    2  RI                  1   1
    12.88 BGN   11  RI                  1   1
    17.88 END   11  RI                  1   1
    17.88 REQ           IDL
    23.97 RDQ           IDL
    23.97 BGN   17  RI                  6   1
    27.83 END   17  RI                  6   1
    27.83 NTR             4     1       6   1
    38.57 BGN    4  RI                  6   1
    49.46 END    4  RI                  6   1
    49.46 BGN   11  RI                  6   1
    54.46 END   11  RI                  6   1
    54.46 BGN   22  RI                  6   1
    58.30 END   22  RI                  6   1
    58.30 REQ           IDL
    87.29 RDQ           IDL
    87.29 BGN   17  RI                 11   1
    90.29 END   17  RI                 11   1
    90.29 BGN    2  RI                 11   1
    99.50 END    2  RI                 11   1
    99.50 BGN   11  RI                 11   1
   104.50 END   11  RI                 11   1
   104.50 REQ           IDL
   110.45 RDQ           IDL
   110.45 BGN   17  RI                 15   1
   115.36 END   17  RI                 15   1
   115.36 NTR             4     1      15   1
   116.76 BGN    2  RI                 15   1
```

Figure 5: Selective Trace for Resource Number 5 Over Entire Simulation