THE BASICS OF GASP II: A TUTORIAL

A. Alan B. Pritsker
Center for Large-Scale Systems
Purdue University
West Lafayette, Indiana 47907

GASP II consists of a set of FORTRAN subprograms organized to assist the systems analyst in performing simulation studies. GASP II formalizes an approach to simulation by specifying common elements of simulation studies and providing subprograms for performing those simulation tasks that are independent of a particular problem. The GASP II world-view involves specifying the status of a system in terms of the entities of the system. The entities have characteristics called attributes. The values assigned to the attributes provide a quantitative picture of the status of the system at any point in time. Relationships between entities are maintained by the membership of entities in files. The status of the system changes when attribute values change or the membership of entities in files is altered. When the status of a system changes, an event is said to occur. The GASP II simulation philosophy is that a dynamic simulation of a system can be obtained by modeling the potential events associated with the system and advancing time from one event to the next event. In simulation parlance this is called a next event or discrete event simulation procedure.

Every GASP II simulation model consists of:

1) A set of event programs that describes a system's operating rules;

2) Lists and matrices that store information; and

3) An executive routine that directs the flow of information and control through the model.

The set of event programs represents the technological logic of the operations of the system being studied. The lists and matrices that store information represent the specific entities and their attributes of the system along with control information to direct the operation of the simulation model. Variables which have been found to be common to many simulation programs are given standard definitions and are referred to as GASP variables. Variables which are problem dependent are defined by the systems analyst and are referred to as non-GASP variables.

For GASP II, the executive routines that direct the flow of information and control are organized to provide seven specific functional capabilities as follows:

1) Event control;

2) Information storage and retrieval;

3) System state initialization;

4) System performance data collection;

5) Program monitoring and event reporting;

6) Statistical computations and report generation; and

7) Random deviate generation.

To obtain a complete simulation program, the systems analyst must write subprograms to initiate the simulation, to define event codes and to simulate the events of the system. Subprograms for all the other functions are provided by GASP II. In addition, GASP II specifies the format for the start and event code definition subroutines. Thus, it is only necessary for the systems analyst to conceive subprograms for the defined events of the system being studied in order to obtain a complete simulation model. In the next section, the subprograms provided by GASP will be described along with definitions of the GASP variables. In the tutorial, examples of the programmer written subprograms will be given to provide illustrations of complete simulation programs written in GASP II.

The GASP II Programming Language

The structure of a typical GASP II program will now be portrayed. The user of GASP II must write a main program that initiates the simulation and whose primary function is to initialize non-GASP variables and to call a subroutine named GASP which performs the executive control function within GASP II. Subroutine GASP's first task is to call subroutine DATAN which initializes the GASP variables either directly or through the reading of data cards. An echo check of the GASP variables is provided. This completes the initialization part of the program. At least one initial event must be established during initialization. The initialization data provides both a picture of the system at the start of the simulation and the initial events required to start the simulation.

Subroutine GASP now switches to its "next event" mode. This involves the determining of the next event to occur and the processing of the event at its appropriate time. To accomplish this, an event file or calendar of coming events is maintained by GASP. This event file is essentially a set of events and their associated attributes stored in chronological order. Each event must have a time of occurrence and an event code associated with it. Subroutine GASP uses the event code to transfer to the appropriate event when that event is scheduled to occur. After the logic and arthmetic manipulations are performed at the event time, control is returned to GASP which then processes the next event. A run is terminated when appropriate switches are set in an event subroutine. (A run can also be terminated based on elapsed time.) When this occurs, GASP calls subroutine SUMRY to obtain a final report describing the statistical quantities collected during the simulation. Subroutine OTPUT is then called so that the systems analyst can obtain any additional output that he desires. Appropriate tests are then made to determine if additional simulations are to be made.

Important GASP II Variables

The following is an alphabetized list of those GASP II variables which are commonly used by the systems analyst in writing a simulation program. All GASP variables appear in COMMON except for the array NSET. The array NSET is used to store all entities and events maintained in the filing system. To allow for changes in the dimension of NSET, and yet avoid recompiling the GASP subprograms, NSET is passed as an argument to most of the GASP subprograms. Complete documentation of the filing system and the use of the array NSET can be found in Reference 1.

Variable Name	Definition
ATRIB(IM)	Buffer storage for attribute values being stored in or retrieved from a file in NSET.
ID	Number of columns in NSET.
IM	Number of attribute rows in NSET.
INN(J)	A code for establishing the ranking for file J.
ISEED	Initial random number.
KRANK(J)	The attribute number on which file J is ranked.
MFA	The column number of the first available column for storing an entry in NSET.
MFE(J)	The column number of the first entry in file J.
MLE(J)	The column number of the last entry in file J.
MSTOP	An indicator for specifying that a simulation run is completed.
MX	The successor pointer row in NSET.
MXX	The predecessor pointer row in NSET.
NCLCT	Number of variables on which statistics based on observations are to be collected during the simulation.
NHIST	Number of histograms that are to be generated during the simulation.
NOQ	Number of files to be contained in NSET.
NQ(J)	The current number of entries in file J.
NSET(I,J)	The filing array containing attribute I of the entry stored in column J. If I equals MX or MXX then the attribute is a pointer to show the relationship of this entry to other entries in NSET.
NSTAT	Number of variables on which statistics are being collected based on time.
SCALE	A scale value to allow fractional values to be stored in the file NSET.
TNOW	Current time for the simulation.

Functional Breakdown of GASP II

A listing of the GASP II subprograms by function is shown in Figure 1. Subroutine GASP is the master control routine and is referred to as the GASP executive. GASP starts the simulation, selects events, sequences time, controls the monitoring of intermediate simulation results, and initiates the printout of the final output when the simulation has been completed. Subroutine GASP is called only by the main program which is written by the systems analyst. Once control is turned over to GASP for a simulation run, control is not returned to the main program until the run is completed.

Subroutine DATAN accomplishes the following general functions:

1) Initializes GASP variables to permit the starting of the simulation;

2) Provides a means of reading the values for the constants used in the various GASP subroutines; and

3) Acts as a vehicle for reading in initial file entries and events which are to occur during the simulation run.

Subroutine DATAN is called only by subroutine GASP.

```
GASP Executive
  SUBROUTINE  GASP(NSET)

Initialization
  SUBROUTINE DATAN(NSET,NA)

Information Storage and Retrieval

  SUBROUTINE SET(JQ,NSET)
  SUBROUTINE FILEM(JQ,NSET)
  SUBROUTINE RMOVE(KCOLL,JQ,NSET)
  SUBROUTINE CANCL(KCOLL,NSET)
  SUBROUTINE FIND(XVAL,MCODE,JQ,JATT,
    KCOL,NSET)

Data Collection
  SUBROUTINE COLCT(XX,N,NSET)
  SUBROUTINE TMST(XX,T,N,NSET)
  SUBROUTINE HISTO(XX,A,W,N,NSET)
  SUBROUTINE CLEAR(NSET)

Statistical Computations and Reporting
  SUBROUTINE PRNTQ(JQ,NSET)
  SUBROUTINE SUMRY(NSET)

Monitoring and Error Reporting
  SUBROUTINE MONTR(NSET)
  SUBROUTINE ERROR(J,NSET)

Random Deviate Generators
  FUNCTION DRAND(ISEED,L)
  FUNCTION UNFRM(A,B,L)
  FUNCTION RNORM(J,L)
  FUNCTION RLOGN(J,L)
  FUNCTION ERLNG(J,L)
  FUNCTION GAMMA(J,L)
  FUNCTION BETA(J,L)
  SUBROUTINE NPOSN(J,NPSSN,L)

Other Support Routines
  FUNCTION SUMQ(JATT,JQ,NSET)
  FUNCTION PRODQ(JATT,JQ,NSET)
```

Fig. 1 Functional Breakdown of the GASP II Subprograms

Subroutine SET initializes the filing array NSET and establishes the pointing system for all columns in NSET. Subroutine FILEM performs the information storage function and inserts the values contained in the buffer storage vector ATRIB into the first available column of the filing array NSET. The pointers associated with this column are then updated so that the new entry in the file points to its correct predecessor and successor based on the ranking attribute, KRANK, and priority rule, INN established for that file. To remove an entry from a file, subroutine RMOVE is used. Arguments to this subroutine are the column to be removed and the file number. The integrity of the file is automatically maintained by updating the pointing system. The values of the entry removed from the file are transferred to the buffer storage vector ATRIB. In order to remove a future scheduled event, a special routine called CANCL is provided. To find specific entries in a file, subroutine FIND is used. This routine examines a particular file to locate an entry whose attributes satisfy stated conditions. Five conditions for a search are currently available as detailed below:

1) Maximum value greater than a specified value;

2) Minimum value greater than a specified value;

3) Maximum value less than a specified value;

4) Minimum value less than a specified value; and

5) Value equal to specified value.

The specified value is the first argument and the type of search (1,2,3,4, or 5) is the second argument of calls to subroutine FIND. The file number and the attribute number are the third and fourth arguments.

Examples of the use of the information storage and retrieval subprograms available in GASP II will now be given. To initialize the filing system the statement CALL SET(1, NSET) is used. To file the attribute values stored in the vector ATRIB into file 1 the statement CALL FILEM(1,NSET) is used. To remove the first entry from file 3, the statement CALL RMOVE(MFE(3),3,NSET) would be used. To find the column in which the next event coded with event code 4 is stored, (file 1 is the event file), the statement CALL FIND(4.0,5,1,2, KCOL, NSET) would be used. The column number of this entry would be inserted into the variable KCOL. To cancel the event the following statement is used: CALL CANCL(KCOL,NSET).

Three subprograms are provided within GASP II to allow the systems analyst to collect data during the course of a simulation. Subroutine COLCT(XX,N,NSET) is called every time a variable defined as the Nth variable on which statistics are collected is observed. The code number of the variable identifies the row in an array that is used for maintaining the values of the observations of this variable. Thus, if the time spent in the system for a customer in a queueing situation is defined as the third variable and we have just observed that his time in the system was 15 time units, the statement to record this value would be CALL COLCT(15.,3,NSET). For variables that are time persistent, the subroutine TMST(XX,T,N,NSET) is used. This routine integrates the Nth variable whose current value is XX up to time T, from the last time there was a change in value for the variable, and accumulates this information with other integrations of this variable. Thus, if the current number of customers in a queueing situation is 4, and this value is about to be changed and if the current time is 12, and we have coded this variable as variable number 1, the appropriate statistical information would be recorded by the following statement: CALL TMST(4.,12.,1,NSET). Histograms of specified variables can be obtained by a call to subroutine HISTO(XX,A,W,N,NSET). In this case, the Nth histogram will be for the variable XX which is placed in an appropriate cell where the lower limit of the second cell is given by the value of A and the width of each cell is W. An example of the use of subroutine HISTO to record the third variable whose value is 12 into a histogram whose second cell lower limit is 2 and whose width of each cell is 4 is accomplished by the following statement: CALL HISTO(12.,2.,4.,3,NSET).

Subroutine CLEAR is used to initialize the arrays used to store the statistical data. Subroutine CLEAR is useful for multiple simulation runs and for simulations involving initial periods in which statistical information should not be included in the final computations.

Subroutines COLCT, TMST and HISTO are used to collect the data that is generated during a simulation. The calculation of statistical values and the printing of the information is performed in subroutine SUMRY. Subroutine SUMRY is the basic output routine of GASP II. For each variable on which statistics are collected, the average, standard deviation, minimum value observed and maximum value observed are printed. For those variables for which statistics are based on observations (as collected in subroutine COLCT) the number of observations is printed. For those variables that are time persistent, the time over which the variable was observed is printed. A tabular histogram of the values recorded by subroutine HISTO is also printed out by subroutine SUMRY.

Subroutine SUMRY calls subroutine PRNTQ(JQ,NSET) which prints out the average number of entries in file JQ, its standard deviation and the maximum number of entries that were simultaneously in file JQ. Also printed are the current entries in file JQ. Subroutine SUMRY and PRNTQ do not change the value of any variable and, therefore, can be called at any time during a simulation. Subroutine SUMRY is normally called at least at the end of a simulation run.

Subroutines MONTR and ERROR assist the systems analyst in debugging and tracing portions of his simulation program. Subroutine MONTR provides the capability to selectively monitor events for debugging or program tracing. To initiate the monitoring of the simulation, a monitor event (code 100) is established which communicates to subroutine GASP that the attributes of each event should be printed. When a second event coded as 100 occurs, the monitoring of events is halted. Subroutine monitor can also be used for printing the entire filing array by establishing a monitor event with code 101 to occur at the time the filing array is to be printed.

Subroutine ERROR(J,NSET) is called when an illogical condition occurs within the GASP program. When subroutine ERROR is called, all pertinent arrays are printed so that the cause for the illogical condition can be diagnosed. Calls to subroutine ERROR can be made directly by the systems analyst and his own code established through the argument to subroutine ERROR.

Random deviate generators for the uniform, normal, lognormal, Erlang, Poisson, gamma and beta distributions are provided within GASP II. Each of these generators uses a random number generator as defined by the Function DRAND(ISEED,L). The user of GASP II must supply this DRAND function which generates pseudo-random numbers. Since pseudo-random number generators are machine dependent, GASP II subprograms call DRAND and it is only necessary to modify the function DRAND when transferring from one machine to another. Different random number streams can be maintained through the use of the argument L.

Two support functions are available with GASP II. Function SUMQ(JATT,JQ,NSET) sums up all the attribute values of attribute JATT in file JQ and returns them to the user as the value of SUMQ. Function PRODQ(JATT,JQ,NSET) multiplies the values of attribute JATT that are in file JQ and returns them as the variable PRODQ. Additional support functions of this type follow the same pattern and can be easily incorporated into a simulation program by the user.

In the tutorial, each of the GASP II subprograms will be described in detail. Examples of the use of GASP II will be presented which will include the programming required, the input data formats and the resulting summary reports.

REFERENCE

1. Pritsker, A. A. B. and Kiviat, P. J., "Simulation with GASP II: A FORTRAN-based Simulation Language", Prentice-Hall, Inc., Englewood Cliffs, N. J., 1969.