A graphics and information retrieval supervisor for simulators

by JAMES L. PARKER

University of British Columbia
Vancouver, Canada

INTEREST IN SIMULATION

Interest in simulation by computer has grown steadily over the last ten years. This interest has become even more strong in the last couple of years with the advent of popular interest in environmental and social systems. Frequently the most limiting factor in the simulation and in the evolution of the techniques of simulation is not the ability to generate the programs to do the arithmetic. It is the ability to handle in an easy way the information flow which results from the simulation and to provide users—particularly those who are not intimately familiar with computers—an easy, versatile, hands-on way of displaying this information.

The above need is felt in two ways. First when a new simulator is written a significant portion of the effort of its design and coding goes into that portion of the code which maintains the data base on which the simulator operates. Another significant portion of effort goes into the system which allows the simulator to communicate with a user. This repetitious diversion of energies and money from the primary interest of the people who write simulators (the arithmetic itself) is at this time severely limiting the growth of simulators. In addition the lack of easy graphics display and information retrieval makes it more difficult to correct and evolve the simulators. One can correct and calibrate a simulation more rapidly if one can see exactly what it is doing and if there is a system which allows a sort of throttle control over the activities of the simulator.

The current limited usefulness of simulators can partially be attributed to the fact that the real people who should use the results of the simulators are people who are not familiar with computing equipment. They frequently have only an intuitive feel for what they would like to display and how they would like to see it. Most simulators currently built are left to wither in the academic realm because there is no mechanism for communicating the powers of these simulators to the people who are really interested.

It is the purpose of this paper to explain an information retrieval and graphics supervisory system which will accept most forms of simulation and execute them as subsystems. It provides users with an easy and flexible way of maintaining the data output from the simulators and of displaying this information in graphic and tabular form.

THE IIPS PROJECT (Inter-Institutional Policy Simulator)

The work that is described in this paper was done with a particular simulation project in mind, even though the system itself is not simulator dependent. This project is the Inter-Institutional Policy Simulator project financed by Ford Foundation and as a joint effort between the University of British Columbia, the City of Vancouver, the Greater Vancouver Regional Board, and the Province of British Columbia. The purpose of this project is to involve several diverse institutions in the development of a simulation which can then be presented to the various governmental and social agencies in the Vancouver region. The project is characterised by having several groups working on diverse simulators which must interact. Also its ultimate audience does not in general have a high familiarity with computers. It is also a requirement with this system that it be independent of the types of devices on which output is to be displayed. This project therefore has made an ideal testing ground for the ideas presented in this paper since the variety of people and the variety of simulators involved makes a maximum test of the design of the system itself.

THE SUPERVISOR FROM THE USER'S POINT OF VIEW

This description from the user's point of view will freely use examples based on the regional project being
done in the Vancouver area. This means that, because this simulation is basically over time and subregions of the study area, the dimensions of time and region will be used liberally in the following descriptions. As far as the simulator supervisor itself is concerned, time is not a dimension any different from any other dimension or key in the information retrieval system. The only exception to this is the fact that the simulation supervisor does assume that there is one dimension in the simulation which is unique in that the policy interventions explained below are over a certain interval of this dimension. In the vast majority of simulations being built today this dimension is time, so there is no confusion. This dimension could just as easily be distance, however. Let's say that one is simulating a rocket trip from the earth to the moon. The dimension chosen might be distance rather than time. It is conceivable for the simulation of an electronic circuit that this unique dimension would be a voltage level. In the following discussion, however, it will always be assumed that this dimension is time over a subset of regions.

THE INFORMATION BASE

The user of this system sees three things. The primary aspect of this system from his point of view is a large data base which represents all of the available information about the system being simulated over a time interval in years which is chosen by the user. As far as the user is concerned all of the data base information is always present in the computer files. He simply has to display whatever aspects of the system are of interest to him.

The second element which the user sees is called the policy vector. This is a set of variables each of which has a name conveniently chosen for the user. The names themselves may be redefined by the user. Each variable can take on different sets of values over time. As an example, the interest rate prevalent in an economic situation would be a single variable which the user might want to set to different values for different subintervals of the time span in which he is interested. These variables which can be set by the user constitute the policy interventions which the user can make into the model. In this way the user can attempt to study the effect of certain policy decisions and to see the differences in the display for different policy values. Each policy variable has a certain default value at the time the user begins a session with the simulator. He can change this default value to any value he chooses over the entire time interval he is studying or over a subinterval. Assume that the time interval is 1970-1990 and assume that the initial value for the interest rate is 7 percent. The user can change this value over the whole time interval or he can change the value for a subinterval. For example, he could make the interest value 8 percent in the interval from 1980-1985. As far as the user is concerned the current state of the data base reflects the set of policy values which he has chosen and they also reflect the total situation over the whole time interval he is studying. He can also set the time interval that he wants to study.

The third element available to the user is the command language with which he requests displays and asks for changes in policy elements. Although these are the two main functions of the command language there are many other commands associated with the type of displays desired, the devices available, and with saving and restoring portions of the information base.

SOME SIMPLE EXAMPLES

Before describing in more detail the structure of the information base or the command language, it might be well to show a couple of examples of the types of displays a user might request. Let us assume that the user is aware that the figures for population for a set of regions in the study area are available in the data base. Assume that there is one population figure for each region and each year and one total figure called POPULATION for each year. In that case, he might make the following simple display statement.

DISPLAY POPULATION BY TIME

This produces a simple graph shown in Figure 1.
Then let us assume that the user gives the command

CHANGE INTEREST RATE TO 8 PERCENT

He then repeats the first command. In this case, he would get the following graph in Figure 2 displaying the population figures reflecting his policy intervention. As in indicated later it is possible to split whatever device he is using for display into several areas. Using this feature, he could have displayed these two graphs one beside the other on his screen or his paper. He could then compare directly the two population figures. It was also possible for him to give the following command as his second display command and he would have the two graphs superimposed over each other. Thus he could compare them on the same area.

DISPLAY POPULATION BY TIME
SUPERIMPOSED

Or he might have given the following single command which would have produced the same results but with each graph marked by the corresponding interest rate:

DISPLAY POPULATION BY TIME
BY INTEREST RATE FROM 7 PERCENT TO 8 PERCENT

This would have produced the graph in Figure 3.

The simple examples give a clue to the power of the system by showing the direct control the user has over the actions of the simulator. It also suggests that there exists a wide variety of types of displays. The following shows how the user relates these displays to whatever devices he has available and what display types are available.

DISPLAY DEVICES

The system is set up so that no matter what sort of device the user is working from he can receive some type of display. The limits of the display for given device type are dependent on the quality of display that the device can support. If the user is sitting at a teletype, he can receive graphs of whatever resolution a typewritten line can support. Similarly, if he has a line printer available to him, he can receive plots on the line printer. If he has a scope available, then he can receive his displays with the much greater resolution and flexibility that a scope allows. Finally if he has a Calcomp, he can receive the same resolution as that of the scope but with the limitations that the use of paper obviously implies. The system as implemented at U. B. C. supports an Adage type non-storage scope, a Technix scope, Calcomp plotters, line printer, 2260 display, teletype or 2741 display, and a dot printer.

The user has quite a bit of flexibility in the way in which he can relate his displays to the particular devices. In the above sample command, the defaults were all in effect so that the user did not have to worry about assigning the graphs to areas. For a particular device, the user is allowed to cut the face of the device into 1, 2, or 4 areas. This is most advantageous when using a Calcomp or storage scope. It is also possible that the user has more than one device available at any given time. If he has for example a scope available and a Calcomp this allows him to develop certain images on the scope and then have them plotted on the Calcomp. In the case that the user has established multiple areas he must indicate the area on which he expects a display to appear. As was evident in the above examples, the user can cause a new graph to be superimposed on an old graph or he can start with a completely fresh image.
in an area. It is also possible in the command language for him to move a current display from one area to another. This allows an evolution of a display on a CRT type device and the creation of hard copy from a device like a Calcomp. In addition to this the graphic supervisor is set up so that it provides a memory buffer of images for each area currently in use. This implies that whatever images have been created for a given area, a user can always step back and review a previous image or move forward and again see the current image. The number of images contained in the memory buffer for each area is defaulted to five but it can be changed by an individual user.

RATES OF DISPLAY

It must be remembered that the whole purpose of this system is to display information quickly to a human being and in a form which he finds understandable and pleasing. In light of this it can be observed that a static display is not in general pleasing to a human being. It is much more intriguing and informative to see something that moves. For this reason, the displays on this system have been given a wide flexibility in the way in which they are displayed with respect to time. Naturally these methods of display with respect to time are much more flexible on a CRT but some of these principles are also applicable to paper graph displays.

Each graph is considered to be a subset of a graph with four dimensions. It may seem rather surprising to consider a graph as having four dimensions. However, it is possible to superimpose several graphs in one area, one already has three dimensions, as in Figure 3. In Figure 3, the three dimensions are population, time and interest rate. Do not confuse time as used here with the idea of the real time of the person using the system. Time here is simply one of the data variables from the information base. If a sequence of figures such as that in Figure 3 could be displayed where each frame in the sequence corresponds to the given value of some fourth variable, the display would effectively be showing the relation of four variables. Not only that, but it would be showing it in a way that would be fairly palatable to human understanding. If the user can see the change in his graphs from one frame to the next, it is actually a fairly good way of transmitting information to a person. The idea of the "rate" in these displays is based on these two ideas—that one can display a four dimensional graph and that the graph can be displayed through time.

THE RATE CLAUSE IN THE DISPLAY COMMAND

The general form of the display commands for graphs is as follows:

\[
\text{DISPLAY (SEQUENCED) BY X (SEQUENCED) BY Y (SEQUENCED) BY Z (SEQUENCED) BY T AT RATE R}
\]

The above example does not include all of the features of the DISPLAY command but includes those which are relevant to the rate of display. It is intended that the SEQUENCED phrase can occur only in one of the four positions. These four positions represent the four dimensions which will appear on the graph. The variables X, Y, Z and T are to be replaced by data names from the data base. The form of the display is determined by which variable has the SEQUENCED phrase associated with it. If the SEQUENCED phrase is associated with the last variable T, then the system merely pauses after each complete frame. Note that it is possible to have less than 3 dimensions in a given display command. The rate R is in tenths of a second. If the word PAUSE is used, then a single frame of the graph is displayed and the next frame is not displayed until the user generates an interrupt on whatever type of terminal he is using. This allows the rate of display to be completely controlled by the user. He can then investigate the characteristics of a single frame before he proceeds to the next frame. Otherwise the image is built up at whatever rate he specifies. The benefit of this type of approach is that it allows one to display more information and to give a person a feeling of something growing. As the graph emerges, one gets the feeling of being able to see a process.

The SEQUENCED phrase can also be associated with any of the earlier dimensions. If it is associated with the Z dimension, then the individual frame of the graph is also built up with pauses in between its parts. Associating the rate with the X dimension also generates a different pattern or growth of the graph and different pattern of pausing. This will be particularly beneficial when using the model for instructional purposes, in which people can discuss a set of policy interventions and then test them out on the model. At this point one can guess what is about to happen in the graph as it grows. After seeing part of the graph one can make predictions as to what will happen next. This will stimulate discussion and the use of the individual’s imagination as to what the simulator might do. For example, a policy intervention might require a period of several years in order to take
effect. This will allow people to stop and discuss their ideas about how long it might be before the policy takes effect. Since one of the primary purposes of a simulation like this is to stimulate people’s imagination, this rate or throttle control on the graph output is very valuable. Figure 4 shows the three different types of growths of graphs which can be displayed. In the succeeding paragraphs, other types of displays than graphs will be explained. Some of these can also have the rate associated with them. At this time no meaning has been associated with the concept of putting the SEQUENCED phrase on the Y dimension.

Lines marked with the same number of cross lines are displayed simultaneously. It is difficult to indicate motion as below by printing on a static sheet of paper.

TYPES OF DISPLAYS

There are actually more types of displays available to the user than those indicated above. Any graph may be superimposed on any other. The five types are:

1. Map
2. Point
3. Line
4. Bar
5. Contour

BAR: The bar graph is of a standard form. It is assumed that if more than two dimensions are expressed in the display statement for a bar graph that the third dimension will indicate different time frames. It does not make sense to have more than three dimensions for a bar graph.

LINE: The line drawing is just the standard line graph which has been explained above. This can have from two to four dimensions.

POINT: If the display command is indicated to be of type POINT, the first two dimensions are taken to be the X,Y coordinates of the point. Point data may or may not be superimposed over a map. If the X and Y coordinates are the coordinates of some regions on a map, then it can be superimposed over a map. There are other cases, for example population versus unemployment over several regions, which would generate a scatter diagram for all of the points within the specified time interval and for all of the regions in the model. In a point graph, if the third dimension is given it is assumed to be a Z value and the numerical value is drawn in on the graph at the spot indicated by the X and Y coordinates. If there is a fourth dimension, it is assumed to always represent different time frames in a point graph.

MAP: Maps are generated from files in the data base which are the output of point digitizing equipment. They are currently all outline maps. Any point data can be superimposed over an outline map.

CONTOUR: This type of display is generated from point data with at least three dimensions. In this case the first two dimensions indicate the location of point on the map as in the point graph and the third dimension indicates the value to be used by the contour generating program. If there is a fourth dimension, it is assumed to represent different time frames as always.

The form of the display command to indicate the type of display is as follows:

DISPLAY/X/BY/Y/BY/Z/BY/T AS MAP
BAR
LINE
POINT
CONTOUR

See Figure 5.

THE COMMAND LANGUAGE

The above examples have given references to various commands in the language. Although it is not the intention of this document to produce a user’s guide to the language, several commands are explained below with their features to give an idea of the range and flexibility of the system.
The commands are broken into three groups: display and data manipulation commands; systems and graph manipulation commands; and user aid and text formatting commands.

The display and data manipulation commands are LIST, DISPLAY, SAVE and RESTORE.

The LIST command has several options. It can list for the user any of the names in the system, such as the names of the policy vector interventions available to the user, the names in the information base, the file names into which the information base is organized, or any text replacement names which the user has defined. The list command can also output data from the information base in tabular form if this is more desirable than having it in graphic form.

The options on the DISPLAY command which have not yet been mentioned are that data may be summarized according to another variable which may or may not be in the display. For example, if population data is present in the information base by region for display purposes it could be summarized by time so that a single value would appear for the total study area for each time interval. Also on any one of the four dimensions, limits can be attached by typing FROM lower limit TO upper limit BY increment, where the increment value is simply the value used to step from the lower limit to the upper limit. Soon there will be added a feature to the system whereby any of the display dimensions can be replaced by an arithmetic expression. Then given unemployment, population and total population available in the data base, one could say 'DISPLAY UNEMPLOYMENT/POPULATION BY TIME,' and this ratio would then be calculated. It is also possible in a display command to introduce a policy vector entry. For example:

DISPLAY POPULATION BY TIME
BY INTEREST RATE FROM 8% TO 9%

It is very likely that the user, after developing a data base and a set of policies in which he may have invested quite a bit of time and money, may want to save all or part of the current status of the system. It is also possible that the user may also want to investigate a decision tree of his own policy strategies and to compare the results. In order to do this he must have the capacity to save all or part of the system in his own private files. This is the function of the SAVE and RESTORE command. The SAVE and RESTORE commands can save the total contents of the system, the contents of any one of the data files, or the symbol names that a user may have defined. See below for an explanation of the user defined symbol names.

SYSTEM AND GRAPH MANIPULATION COMMANDS

These commands set up the structure of a given session using the simulator supervisor and alter the status of the system. The ESTABLISH command initially indicates which devices are available to the system and how many areas each device should be broken into. This allows the use of multiple devices of different types simultaneously. For example, in one location in which the system is used, there is both a storage scope and Calcomp plotter available. Since each area has a memory buffer, there must be commands for displaying images stored in this buffer. These are the GO BACK and GO FORWARD commands. These commands will also transfer an image from one area to another. This allows one to take a display from the memory buffer of a scope device and to display it on hard copy such as a Calcomp.

The ERASE command simply blanks the image on one area. The primary use comes when the system is being used for teaching purposes and one wants to have the viewer's attention directed elsewhere.

The SET command changes a variety of different system functions and defaults. This is used to set the number of images contained in the memory buffers, certain defaults about the way files are to be handled and so forth.
USER AIDS AND TEXT HANDLING COMMANDS

The EXPLAIN command can be followed by any of a variety of names. If it is a name of an item in the data base, a prewritten explanation of this item will appear. If it is a command name, a portion of the command manual will appear. If an error has occurred and the command says "EXPLAIN ERROR", a prewritten explanation of the error message will be printed out.

The HELP command currently supplies the user with information about where he can obtain various types of information about the system. It is hoped that this function will be greatly expanded in the future.

It is possible for the user to define any character string as any other character string and to have this text replacement occur in his command. The ability to redefine text allows each user to build up during the session the environment in which he operates. This means that he can redefine any name in the system to be a name more comfortable to his own way of thinking. He can also give names to graph descriptions which he feels he wants to display frequently. This will be particularly useful when users begin to generate arithmetic expressions which they do not want to repeatedly type out. The commands here are DEFINE, which defines one string as another string for replacement purposes, i.e.,

DEFINE POP AS POPULATION_VALUES

This would allow the user to use the characters POP whenever the full name was required. DROP followed by a string simply drops that string from the replacement table.

DROP POP

It will frequently be the case that a user discovers that he wants to name a whole description of a graph. So it is very convenient to have this built right into the display command itself. The command is then DISPLAY followed by the new name followed by the word AS followed by the usual graph description. This enters the name given in the replacement table and enters as its replacement the string describing graph as below.

DISPLAY P_GRAPH AS POPULATION BY TIME BY REGION AT RATE 40

THE SUPERVISOR FROM THE SIMULATORS POINT OF VIEW

One of the primary goals of this system is to make it very easy to add a new simulator and to minimize the burden of input/output command parsing and display on these simulators. For this reason the world into which the simulator fits is made extremely simple. As far as the simulator is concerned, the communication with the outside world consists of reading variables from files which are accessed by a file number and writing other numbers into other files. In case of an error there is a routine which is called with an error message number and which has some macro capabilities. The files that the simulator sees are basically sequential files with keys allowing either a sequential or random read or write. The file descriptions which define the names of the values which will be put in these values are written out in simple character format very much like the data description for any high level language. These file formats are shared by all the simulators in the system at one time, so that one simulator can read the files generated by another simulator. Since these file descriptions are in character format, they can be easily changed and evolved as the simulators themselves evolve. It is only required that (as in FORTRAN style input and output) the simulator assumes the function of each variable in each row that it reads in from the file. For one given file, the simulator must be aware when it is written that the fourth variable of this file is population.

MORE DETAIL ON THE INFORMATION RETRIEVAL SYSTEM

The information retrieval system is really the heart of the simulator supervisor. It must be able to access the files by key and it must be able to support a fairly complicated key structure within the files. For example, there will be files which are indexed by time and region and there will be some files which will be indexed only by region. Yet for display it would be necessary to relate these two files by region alone ignoring the time key on one of the files. There would also be files by time and region and other files by time region, and time of day. This effectively means that the information system must support a hierarchical data structure with multiple keys. However, the vast majority of accesses to the data base by the simulators are made in a completely sequential fashion. That is, very little of the structure is used in a hierarchical fashion. The possibility must be there when it is required.
In order to maintain this facility, the information retrieval system has been structured to operate on sequential files with fixed length records. These are designed to look like FORTRAN vectors; they are just a row of variables. In order to maintain the facility of multiple keys, each file may have several keys which are concatenated together to form a single key. If one wants to generate a facility for accessing the kind of structure mentioned above where one file is by time and another by time and region, the implementation must have two files with the information recorded by key in each file. To do this key lists are kept for each file which gives the key value and a line number or record number for locating the corresponding record.

Since it is possible to change policy variables over certain intervals of time it is also necessary that there be a mechanism for invalidating portions of the data base. For example, if the current data base had been computed from 1970 to 1990 and the interest rate were changed in 1980, it would be assumed that some of the output from some of the simulators would no longer be correct if they were influenced by the interest rate. Therefore those files would have to be erased beginning with 1980. This can be accomplished by simply erasing the elements in the key list which have year values after 1980.

This organization of the information retrieval system allows very simple access routines for the simulator. Yet it will allow a fairly complex hierarchical structure of keys if necessary. The fact that the user assumes all the data to be present at a given time and that the data is actually computed by the simulators only when it is requested for a display means that unnecessary computations never have to be made. Therefore the information retrieval system must be able to communicate to the supervisor what data is actually present in a file at a given time.

In order to explain to the system the structure of the files, and the interdependencies among the simulators for data, there must be a set of simulator descriptions and the file descriptions. These file descriptions as mentioned above are simply lists of the variable names that will be used by the user for referencing the data. It also includes other characteristics of the variables such as the maximum value that they should be allowed to obtain and the minimum. For each file there is a description of the simulator which writes the file and how to load that simulator in order to execute it. There is also a set of simulator descriptions which describe the files each simulator reads. This is sufficient information for the supervisor to decide on a given display command what data is required, what data is missing, which simulators must be called to generate the data and in what order the simulators must be called. The rule is that a single file may be written by only one simulator. This is required so that the supervisor can tell how to replace data which has been invalidated by policy changes or which may never have been computed in the first place. In order to generate missing data, the supervisor steps through the years in the current time interval and at each year runs the simulators required to generate data for that year. These simulators are dynamically loaded and are kept in core as long as possible until another simulator is required to run.

AN OVERALL FLOW DIAGRAM OF THE SYSTEM

The system is broken up into three major sections of code. The primary section is the supervisor itself. This code does the initialization work of reading in all the tables, reads in each command from the user, and converts the alphabetic strings in the command to integers which are merely reference numbers corresponding to the strings. This allows command interpretation to be done very easily. It is at this point that text replacement is also done. The commands then go through a large switch to individual subsections of the supervisor which process each command. The second section of coding are the I/O routines. These routines maintain the key lists for the individual files and do all of the sequential and random reading and writing requested either by the simulator or by the supervisor.

At the time the simulator originally begins running it is assumed that each file that is non-empty is in a read-only master copy. As long as display commands are generated for these files this read-only copy is used. As soon as a simulator wants to write a file a scratch copy is made. This copy can either default to a temporary file or can be put into a private file of the user, depending on a systems switch. Notice that the I/O routines, except for saving original data, treat observed data and simulator output exactly the same. There is no distinction between data bases which were...

Figure 6—Overall system structure
originally observed and which are modified by simulators, data bases which were written only by simulators, and data bases which were only observed. This symmetry of treatment coupled with the ability to describe files in format-like statements allows the structure of the files to evolve very easily.

The third piece of code involved is the graphics supervisor. This section of code maintains the current state of all of the display devices in the system. It also maintains the image buffers, the memories for each area of each display. It is the responsibility of this section of code to convert a standardized graph description which it receives from the supervisor into the proper display for each device in the system. It is also the responsibility of this section of code to OR the graph images together into single images when superposition has been requested. This part of the program also does the computations required for contouring and displaying of bar graphs and so forth.

EXTENSIONS

The most exciting extensions to this system will be in the area of providing facilities to the user for developing policy strategies. The first cut in this area will be to give him the ability to develop a decision tree of alternatives, and to compare and display data across these different alternatives. See Figure 7. At this point it would also be of value to be able to provide him an analysis of which variables in the data base are affected and at what level by changes in the policy variables. At first this can be done on a guesswork basis by knowing which data is invalidated by changes in which policy variables. Later on, however, it may be interesting to do more in-depth analysis by actually running the simulators for a short period, noting which variables change, and attempting to give some indication of the derivatives involved.

What would also be very interesting in this context is to generate a rudimentary artificial intelligence device which searches out policies which tend to satisfy certain goals set up by a user. In this case the interaction between the user's intuition in suggesting goals and a rudimentary artificial intelligence device might produce some very exciting results.

CONCLUSION

It is strongly believed that this system will bring the power of large interactive computer systems rapidly to users in two different ways. Primarily, it will provide an ability for an untrained user to investigate in great detail and according to his own experience the workings of a set of simulators. Secondly, by relieving the simulator writers of the burden of providing user interface and information retrieval it will allow the quality of the simulators and the types of the simulators themselves to be evolved very rapidly.