SIMDABS: A DATABASE SYSTEM TAILORED FOR USE IN SIMULATION STUDIES

Charles R. Standridge
A. Alan B. Pritsker

ABSTRACT

The SIMulation DAtabase System (SIMDABS) is a FORTRAN based data management system designed for use in simulation studies. While SIMDABS provides several commands for general data processing, most commands are tailored for the data manipulation needs of simulation. Commands may be accessed from any computer program which is able to call FORTRAN subprograms or through the use of the SIMDABS Operation Invocation Language (OIL). SIMDABS not only provides data handling commands, but also provides a data model, or framework, which guides the user in the organization of his data.

INTRODUCTION

While many simulation languages have been developed to facilitate the modeling of systems, little attention has been paid to aiding the modeler in handling the data associated with his study. Until recently, as Miller and Morgan (7) point out, the only tool available to the simulation modeler for data handling was the file manipulation system of the computer he was using. Babad and Schrage (1) suggested the development of data handling methods oriented toward process-oriented simulation languages to improve debugging facilities, to allow the batching of individual observations, to save the system status, and to facilitate report generation.

SIMDABS aids the simulation modeler in the organization and manipulation of all data associated with a simulation study. It can be used for organizing, storing and retrieving simulation input data and output results. SIMDABS extends simulation modeling capabilities by providing user-callable functions for organizing, gathering, analyzing, and retrieving data from within simulation models. These capabilities provide methods for implementing data dependent, complex logic within simulation models.

SIMDABS consists of four components:

1. the data model;
2. the data management functions;
3. the operation invocation language; and
4. the physical data management system.

In this paper, each of these components will be presented in turn. First, we provide a review of the uses of database systems in simulation.

DATABASE SYSTEMS IN SIMULATION

In several studies, existing general purpose database systems have been incorporated into simulation languages for a specific purpose. One example of this is the development of INDECS, a general conveyorized facilities simulation package, by Nof and Wilson (8). A computer program written in the GASP II simulation language is used to simulate the system. An ADBMS database was used to hold model parameters such as the exact configuration to be simulated. Another example by Joseph and Roberts (4) involved the integration of the GPLAN database system into the INS simulation language. The INS analysis program causes the detailed output of a simulation run to be stored in a GPLAN database. Thus, instead of being limited to summary statistics, the user is given access to the step-by-step information generated by the simulation of his system. A somewhat similar approach was taken by Duket and Wortman (3) who stored the output of a SAINT simulation in a computer file. Then, the Statistical Package for the Social Sciences was used to produce the desired analyses.

Markowitz (6) describes a database system based on an entity-attribute-set data model which has been incorporated into SIMSCRIPT. By designing the data model in accordance with the representation of model entities, these entities, their attributes, and their relationships can be directly stored in a database. Markowitz proposes the entity-attribute-set model as a flexible, general framework for organizing data.

THE DATA MODEL

A data model is a framework into which data must be organized to be handled by a database system. The purpose of the data model is to specify the logical structure in which the user should conceive and express manipulations of the data (5). Thus, a manipulation need not be expressed according to the physical organization of the data required by a computer system.

SIMDABS uses a relational data model (2) which consists of a set of relations. A relation is a two dimensional tabular form with a specified number of columns (variables) and any number of rows. All information stored in relational form is represented by data values. That is, no relationship between data elements is imposed by the data model.

Two modifications to the basic relational data model have been incorporated into SIMDABS. The first modification allows the user to partition each relation into one or more sets of rows called tables. This partitioning is useful in the case of multiple runs where data concerning the i^{th} run can be associated with the i^{th} table of a relation. Tables are also useful when a system contains multiple occurrences of a single component, and the data associated with each occurrence is to be kept distinct. For example, the outputs of a population projection model for a state, desired by county, could be stored in tables of the same relation, with each table corresponding to a county.

The second modification allows the declaration of one column of a relation to be a key column. Within each table of a relation for which a key column has been specified, no two rows may have the same value in that designated column. When using this capability, rows must be loaded in ascending order of key column value. This feature of SIMDABS facilitates the construction of indexes which allow for the quick retrieval of any particular row of the database. This type of retrieval is the foundation for several of the abilities of SIMDABS which increase simulation modeling capability.

THE DATA MANAGEMENT FUNCTIONS

Data management functions provide the user with a framework for organizing, storing and retrieving data as well as a set of operations for processing, analyzing and presenting data. The logical data management functions of SIMDABS are grouped into five types: definition, general data processing, statistical collection, accessing statistics within models, and statistical analysis of simulation output. Each of these types is discussed below.

Definition. The definition commands allow the user to describe the relations he needs in his database and the manner in which the non-definitional data management functions are to be performed. Through the use of definition commands, clear documentation of defined relations and data management operations is automatically provided. Definitions of relations and data management function actions are stored in a SIMDABS database and retrieved when referenced.

General Data Processing. The general data processing functions are used to perform data storage, retrieval, and presentation operations. These operations are performed in accordance with the definition commands described in the previous section.

SIMDABS provides several options for retrieving one row of data from a table of a relation. If a relation has a key column, a row having a specified key value may be retrieved. Alternatively, the current row of a table may be returned where the current row of a table is the last row accessed by SIMDABS. In addition, a command allows the current row of a table to be established by the user. Another option allows the retrieval of the next row, that is, the row following the current row. Finally, the row having a specified address in the database may be retrieved. A function is included for finding such addresses. These options are referred to as retrieving according to: key value, current row, next row, and database address. Similar options are available to put a row in the database.

SIMDABS provides five ways to retrieve data in a modified form. Modified retrieval allows the user to select particular columns or rows of data. The batching of rows of data may be performed, and corresponding rows of several tables may be concatenated into one row or batched into one row. The correspondence between rows may be defined by either a position in a table or by key value. Combinations of the above forms of modified retrieval may be used. Modified retrieval is a convenient way to create input to other data management functions or to create new relations belonging to a database.

Other general data processing functions included in SIMDABS are: Reporting of data; graphing of data; creating input files for non-SIMDABS programs; transferring data from a computer file into a relation of a SIMDABS database; and checking the range of data in a relation and correcting rows found to contain range errors.

Statistical Collection. The collection of statistics is an important part of most simulation model programs. In SIMDABS, the user, through definition commands, describes the statistical computation procedures.

Statistics are computed on groups of rows. The number of rows in a group can be specified as an integer value. Alternatively, the last row in a group may be indicated by a designated value in a specified column. Weighted statistics may be obtained by defining one column as containing weighting values for the other columns. Groups of rows may be analyzed independently of other groups or cumulatively with other groups. Rows may be formed using modified retrieval.

Relations to hold the results of statistical computation are automatically defined by SIMDABS. For each column to be analyzed the mean, standard deviation, count, maximum and minimum are computed. Histograms are obtained if requested.

Five functions implement the statistical collection component of SIMDABS. These functions provide options for analyzing data stored in the database, for analyzing data not stored in the database, for reporting statistics and for changing the form of statistical computations stored in the database.

Accessing Statistics Within Models. In addition to providing a flexible format for the gathering of statistics, SIMDABS provides commands for accessing statistics during a simulation model run. The commands provide for the following capabilities: 1. Return all of the statistics which have been computed concerning the values in one column from one group of rows; 2. Retrieve a histogram, in cumulative distribution function form, built from the values in one column from one group of rows;

3. Retrieve one specified statistic concerning all columns from one group of rows; and 4. Return one statistic concerning one column which stands in a specified relationship to the same statistic concerning all other columns computed from a group of rows. The functions associated with the above commands can be used to access statistics from past runs or a current model run.

Statistical Analysis of Simulation Output. SIMDABS provides three commands for the computation of the variance of the sample mean from data stored in a database. In using any of these functions, the user may specify, as previously stated, the use of modified retrieval in creating the rows to be analyzed and a column from which weighting values will be taken. All relations needed to hold the results of the analysis and reports concerning the analysis are generated by SIMDABS.

The first of these commands employs the replication technique to analyze data gathered from multiple runs of a simulation program. In this mode, the user stores all data in the same relation with data concerning the i^{th} run stored in the i^{th} table. The report generated by the command gives statistics for each run for each column and over all runs. Provision is made for the deletion of rows which may contain biased samples.

The second command uses the technique of subinterval analysis to analyze data gathered from a single run. The user specifies the number of observations in each subinterval and row truncation conditions. The report generated by this command gives statistics for each subinterval and over all subintervals for each column specified.

The third command uses the technique of regeneration to analyze the data gathered from one run of a simulation program. The user specifies the value in which column that signals the end of a regeneration cycle. The function computes and reports statistics employing the jackknife computation method for each specified column over all regeneration cycles.

THE OPERATION INVOCATION LANGUAGE

All of the data management functions may be used from a simulation model program or any other program capable of calling FORTRAN subprograms. In addition, SIMDABS has an operation invocation language (OIL) to allow the definition, processing, analysis, and presentation commands to be employed in a stand alone manner. The format of an OIL statement is:

function name,parameter$_1$=value$_1$,...,
parameter$_n$=value$_n$;

Parameters may be specified in any order. Default values are prescribed for most parameters. Therefore, only non-default values need to be specified.

As an example consider the following OIL statement that defines a relation:

```
DFRLT,IDRLT=100,NAME=EXMPL,NCLM=6,NTBL=1,
   ISNDX=YES,IKCLM=1 ;
```

This statement defines relation number 100 whose name is EXMPL. The relation has six columns, one table, and an index based on values in column one.

THE PHYSICAL DATA MANAGEMENT SYSTEM

The physical data management system of SIMDABS provides a mapping from the data model to the structure used for storing the data on the host computer. The user of SIMDABS need not be concerned with the workings of the physical data management system. For a description of this system, the reader is referred to Standridge (9).

SUMMARY

SIMDABS is a tool for the organization and manipulation of all data associated with a simulation study. The SIMDABS data model provides a framework which guides the user in the organization of his data. The data management functions provide commands not only for storing and retrieving data but for describing the operation of other functions, and gathering of statistics in a flexible format, the return of statistics to simulation model programs, and the statistical analysis of simulation output. In addition, reports are printed which present the data stored in a SIMDABS database and which document both the relations which comprise a SIMDABS database and the procedures by which data management functions are performed. SIMDABS is simulation language independent and may be used in conjunction with any program from which FORTRAN subprograms can be invoked.

REFERENCES

1. Babad, Jair M. and Linus Schrage, "A New Look at Process Oriented Simulation Language," Graduate School of Business, University of Chicago.
2. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Volume 13, Number 6, June 1970.
3. Duket, Steven and David Wortman, "An Example to Illustrate the Use of SPSS for the Analysis of a SAINT Model," report to the Aerospace Medical Division, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio, by Pritsker & Associates, Inc., West Lafayette, Indiana, July 1976.
4. Joseph, Pamela E. and Stephen D. Roberts, "Utility of Data-Base Management to Analyze the Output from Complex Simulations," Proceedings, 1977 Winter Simulation Conference, December 1977.
5. Ledgard, Henry F. and Robert W. Taylor, "Two Views of Data Abstraction," Communication of the ACM, Volume 20, Number 6, June 1977.
6. Markowitz, Harry M., "An Entity, Attribute, Set and Event View of Data Base Systems," IBM Research Report RC6811(#29158), Yorktown Heights, New York, October 1977.
7. Miller, Louis W. and Howard L. Morgan, "Simulation Language Features in 1976: Existing and Needed," Proceedings, 1976 Winter Simulation Conference, December 1976.
8. Nof, Shimon Y. and Richard C. Wilson, "INDECS: General Conveyorized Facilities Description and Simulation," Proceedings, 1976 Winter Simulation Conference, December 1976.
9. Standridge, Charles R., Incorporation of Database System Procedures into Simulation Languages, Ph.D. dissertation, School of Industrial Engineering, Purdue University, December 1978.