1986

A GRAPH-BASED APPROACH TO MODEL MANAGEMENT

Ting Peng-Liang
University of Illinois at Urbana-Champaign

A GRAPH-BASED APPROACH TO MODEL MANAGEMENT

Ting Peng-Liang
Department of Accountancy
University of Illinois at Urbana-Champaign

ABSTRACT

A graph-based framework for model management system design is proposed in this paper. The framework applies graph theory to the development of a knowledge-based model management system, which has the capability of integrating existing models in the model base to support ad hoc decision making. In other words, models in the model base are not only stand-alone models but also building blocks for creating integrated models. This guarantees effective utilization of developed models and promises future development of an automated modeling system.

In the framework, nodes and edges are used to represent sets of data attributes and sets of functions for converting a set of data from one format to another respectively. A basic model is defined as a combination of two nodes, one input node and one output node, and an edge connecting the two nodes. A model graph, which is composed of basic models, is a graph representing all possible alternatives for producing the requested information. Each path in a model graph is a model for producing the information. If the path includes more than one basic model, it represents an integrated model. Based on the graphical representation, an inference mechanism for model integration and strategies for model selection are presented.

INTRODUCTION

Graphs are powerful tools for constructing models and solving problems having to do with discrete arrangement of objects. Because of their elegance and simplicity, graphs have been applied in many areas, including economics, operations research, physics, cybernetics, computer science, and other areas in engineering (Busacker and Saaty, 1965). Based on simple idea of nodes interconnected by edges, graph theory combines these basic ingredients into a rich diversity of forms with flexible properties and has contributed to the analysis of a wide variety of systems, such as network flow problems, markov chains, and PERT and related techniques in operations research.

This article discusses the application of graph theory to model management system design. A graph-based framework and mechanisms for integrating models are developed. Because models are knowledge intensive and composed of many functions for converting data, the development of model management systems has traditionally been considered as a difficult research problem. Successful implementation of the framework has not only significantly contributed to our knowledge of model management but also promised more effective use of computerized models.

The remainder of this paper is organized as follows: first, background of model management systems will be briefly described. Then, the

graph-based approach to model management will be discussed in terms of the representation of models, mechanisms for formulating a model graph, and strategies for selecting models in the model graph. A model graph is an inference graph that indicates all possible alternatives for producing the desired information. Finally, an implementation of the proposed approach will be briefly presented.

THE PROBLEM

A model is an abstraction of a specific problem or a class of problems. Because of human cognitive limitations, such as limited short-term memory and bounded rationality, people usually use models to help them understand, organize, study, and solve problems (Simon, 1981). Most models designed to support today's human decision-making are complicated, knowledge-intensive, and implemented on computers.

A model base is a collection of those computerized models. In general, a model base is both integrated and shared. By "integrated" we mean that the model base may be thought of as a unification of many otherwise distinct models with redundancy among those models partially or wholly eliminated. By "shared" we mean that any individual model in the model base may be accessed by any authorized users.

A model management system (MMS) is a software system that handles all access to the model base and provides information to users on demand. Early works in MMSs focused on introducing the concept of model management and proposed that an MMS must support the following functions (Sprague and Carlson, 1982; Sprague and Watson,1975; Will, 1975):

1. creation of new models-- providing an environment to support the model builder so that models can be developed with minimum effort.

2. storage of existing models-- maintaining a model base in which decision models are stored.

3. access and retrieval of existing models-- facilitating the utilization of decision models in the model base.

4. execution of existing models-- executing an existing model and reporting outputs of the model.

5. maintenance of existing models-- supporting the update and modification of the existing models in the model base.

Although these traditional functions may suffice for the need of model management in some systems, they are not sufficient for ad hoc decision support systems which are concerned with problems that are not usually anticipated or recurring (Donovan and Madnick, 1977).

For example, a model base has an economic order quantity (EOQ) model, which computes the EOQ for a specific year from the demand, holding cost, and ordering cost for the year, and three demand forecasting models which employ the regression, demand function, and moving average approaches to forecast future demand respectively. Suppose a user needs to know the economic order quantity for 1987, but he does not have information about the demand for 1987, a required input to the EOQ model. There are two ways for the user to produce the desired output in a traditional MMS: first, create a new model that has functions for both demand forecasting and EOQ computation; second, manually go through the following process:

1. search the model base and find those demand forecasting models and the EOQ model;

2. select one among those available demand forecasting models, and get the input data required for executing the model;

3. execute the selected demand forecasting model and then feed the forecasted demand to the EOQ model;

4. execute the EOQ model to produce the desired information.

The former approach needs effort to create a redundant model, which is the integration of two existing models, while the latter approach needs effort to integrate models by the user manually. Neither has used existing models effectively. Although this example is straightfor-

ward and can be solved easily by the user or any model builder, it does indicate that a powerful MMS needs consulting capabilities which provide advice regarding effective utilization of models in the model base, in addition to those basic housekeeping functions. Namely, it needs the following two capabilities:

1. **Model integration** -- a mechanism for integrating existing models so that each model in the model base is not only a stand-alone model but also a module for creating ad hoc models which are built in case of need. In this example, the MMS must be able to integrate the demand forecasting models and the EOQ model automatically.

2. **Model selection** -- a mechanism that figures out what models are available to produce the requested information and then automatically selects or allows the user to select a model for execution. In this case, the MMS must be able to inform the user of all available alternatives and allow the user to select one for execution or to execute more than one model and then compare the results.

Although research in MMSs has increased dramatically in the past decade, most of it was concentrated on the application of existing data models, such as the relational model (Blanning, 1982-1985) and the network model (Stohr and Tanniru, 1980), or artificial intelligence techniques, especially knowledge representation schemes (Bonczek, *et.al.*, 1980, 1982; Dolk and Konsynski, 1982, 1984; Elam, *et.al.*, 1979, 1981; Watson, 1983).

Few mechanisms for integrating and selecting models in the model base and for providing advice to users have been developed. Recently, Geoffrion proposed an approach called structured modeling, which is focused on exploring functional relationships among the modules constituting a model during the modeling process (Geoffrion, 1985). Although this approach may have significant impacts on the development of MMSs, it is not specifically developed for model management. Nor does the approach provide a mechanism for model integration or model selection in case more than one model is available to support a specific decision.

In order to build the capabilities of model integration and model selection in MMSs, a graph-based framework is developed in this paper. The framework covers three essential issues:

- graphical representation of models,
- mechanism for integrating models,
- strategies for selecting models.

GRAPHICAL REPRESENTATION OF MODELS

In order to build the consulting capabilities in MMSs, the first issue to be considered is the development of a compatible scheme that logically represents the models stored in the model base. Although there may be many different representation schemes, directed graphs are employed in the framework because of their appropriateness, simplicity, and the theoretical soundness of graph theory.

As problem solving is often described as a search through a vast maze of possibilities (Simon, 1981), so can the process of human modeling be described as a search through a number of possible relationships in order to find a route which can convert the initial state (available information) of a problem to the desired final state (output information). By this concept, models in the model base can be represented by two basic elements: nodes and edges. The modeling process can be formulated as a process that creates a directed graph and selects a path in the graph. The directed graph, called a model graph, represents all possible alternatives for solving the problem; each path in the graph represents a model. They can be defined as follows.

<<Definition 1>> Nodes

> A node, N, represents a set of data attributes. It could be the inputs or the outputs of a set of models.

[Example] In Figure 1a, node A represents a set of data including the demand, handling cost, and ordering cost. Node B represents the computed economic order quantity.

<<Definition 2>> Edges

> An edge, E, represents a set of functions that convert a set of input

data (the starting node of the edge) to their associated output (the ending node).

[Example] The edge e in Figure 1a represents the function which computes EOQ from the demand, holding cost, and ordering cost.

<<Definition 3>> Connectivity

Two nodes are connected if there exists at least one edge that converts the data in one node to that in another.

[Example] Node A and B in Figure 1a are connected because edge e converts the demand, holding cost, and ordering cost in node A to the EOQ in node B.

In practical applications, both nodes and edges should be nonempty sets. A combination of two connected nodes and one edge connecting the two nodes constitutes a basic model, the smallest unit in the model base.

<<Definition 4>> Basic models

A basic model, Mb, is a combination of two nodes and an edge connecting the two nodes. The starting node of the edge represents the inputs of the basic model, and the ending node of the edge represents the outputs of the basic model. Hence, a basic model is a three-tuple, <N1,E,N2>.

[Example] The combination of <A,e,B> in Figure 1a is a basic model.

Each basic model in the model base is a standalone model, but it is also a basic element for automatic modeling. Since there is usually more than one way to convert a set of inputs to a set of outputs, the edge between two nodes may not be unique, i.e., more than one model may be available in the model base for solving a specific problem. For example, if one wants to forecast demand for the next year based on the demand data in the last 15 years, one can use the moving average, exponential smoothing, regression, or the Box-Jenkins approach, as illustrated in Figure 1b. In other words, four basic demand forecasting models in the model base, <C,a,D>, <C,b,D>, <C,c,D>, and <C,d,D>, are available for forecasting the future demand.

In addition to the case that more than one model is available to produce a set of required outputs (i.e., there exists more than one basic model which can produce the entire required outputs), it is possible that a set of basic models, in combination, produce the required outputs, but each individual model produces only a subset of the required outputs. In order to differentiate these two situations, we need to define two types of nodes: AND nodes and OR nodes.

<<Definition 5>> AND nodes

An AND node, Na, is a node that is the ending node of more than one basic model; each model produces a subset of the required output, but the whole set of models, in combination, produces the required outputs. An AND node is true only if all models ending at the node are true.

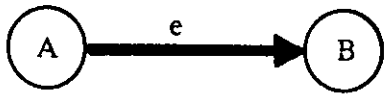
[Example] Node D in Figure 1c is an AND node because the model <A,a,D> produces the demand information, the model <B,b,D> produces the holding cost, and the model <C,c,D> produces the ordering cost. Therefore, the three models, in combination, produce the information contained in node D, but each model produces only a subset of the information. In a model graph, a circle represents an AND node.

<<Definition 6>> OR nodes

An OR node is a node that is the ending node of more than one basic model; each model produces the entire set of required information. An OR node is true if one or more of the model ending at the node is true. In a model graph, a square represents an OR node.

[Example] Node D in Figure 1b is an OR node because there are four models ending at node D, each of which can produce the forecasted demand.

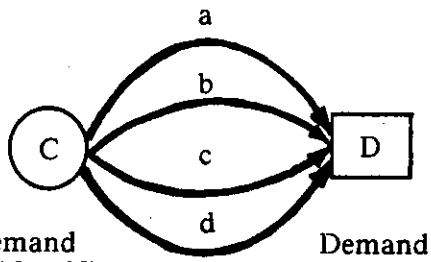
In the human modeling process, an OR node represents a selection point where one or more models are selected among those available models and an AND node represents a union point where more than one set of output data is combined to formulate the required output.



Demand,
Holding Cost
Ordering Cost

EOQ

1a. Graphical Representation of the EOQ Model

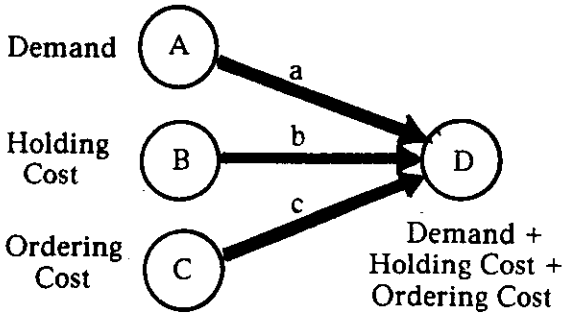


Demand
(N-15,...,N)

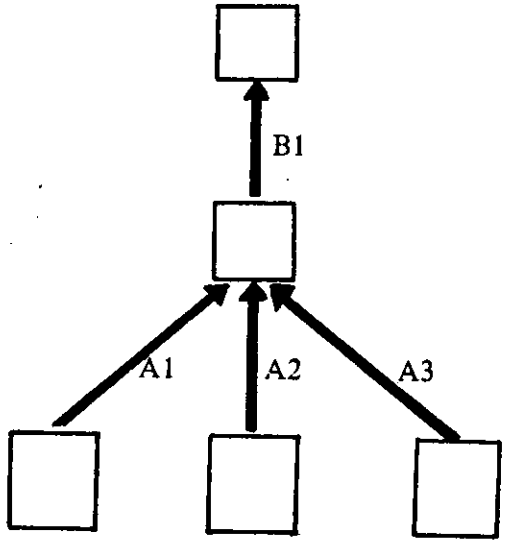
Demand
(N+1)

where: a: moving average
b: exponential smoothing
c: regression
d: Box-Jenkins

1b. A One-Stage Modeling Process



1c. An Example of AND nodes



Where: A1: moving average
A2: regression
A3: demand function
B1: EOQ

1d. A Two-Stage Modeling Process

Figure 1. Graphical Representations

Because all of the four forecasting models represented in Figure 1b provide the same function, i.e., they produce the same set of outputs, and no output of a model becomes an input of another model in the graph, it can be defined as a one-stage graph. However, not all modeling problems are as simple as this example. Many problems may need integration of various kinds of models. By "integration," we mean the output of a model is a subset of the input of another model. For example, Figure 1d is a two-stage graph, which represents models for solving the EOQ and demand forecasting problem described in the previous section. Because the model base has one model for EOQ computation and three models for demand forecasting, there are three paths (1*3), i.e., three different integrated models, for producing the desired information. Formal definitions of the integratibility and integrated model are as follows:

<<Definition 7>> Integratability

Two basic models are integratable if the inputs of one of them are a subset of the outputs of the other.

<<Definition 8>> Integrated models

An integrated model, Mi, is an integration of a set of integratable models. According to these definitions, the concept of a model graph and the modeling process can also be defined.

<<Definition 9>> A model graph

A model graph, G, is a graph that represents all possible models, including basic models and integrated models, for producing the requested information. Each path in a model graph represents a model.

[Example] Figure 1d is a model graph which represents models for computing future EOQ. The model graph is composed of three integrated models. For example, path A1-B1 is the model which forecasts demand by using the moving average technique (edge A1) and then computes the EOQ by using the EOQ model (edge B1).

<<Definition 10>> The modeling process

The modeling process is a process that includes two phases: the formulation of a model graph and the selection of one or more paths in the formulated model graph.

The modeling process is a logical process for formulating the model graph that indicates all possible approaches to producing the requested outputs from basic models stored in the model base. Each path in the graph implies an appropriate model, but it does not guarantee that the model will generate a feasible solution. For example, if a model base contains a capital budgeting model that uses the integer programming technique to determine the best combination of projects for investment, the model graph only indicates the existence of this model. It will not be able to tell the user whether the model can produce a feasible solution until the model is actually executed.

In addition to the formulation of model graphs, an MMS also needs a process for executing the selected path.

<<Definition 11>> The execution process

The model execution process is a process that activates a path and then executes the models constituting the path in an appropriate sequence in order to generate the desired output.

Based on the model graph, an MMS may perform model integration and selection automatically (called the automatic modeling mode) or provide advice and operations for integrating those models to the user and then allow the user to create the integrated model (called the user-assisted modeling mode). Because the model graph clearly represents the relationships among the basic models constituting an integrated model, it becomes much easier for an MMS to provide advice regarding model integration and selection to its users.

IMPLEMENTATION OF THE GRAPHICAL REPRESENTATION

Concerning the implementation of the graphical representation of models, the following five categories of information are essential:

- the output of the model,

- the input required to produce the output,

- the computational procedures used in the model.

- the integrity constraint of the model,

- the validity of the model,

Therefore, in an MMS, a basic model can be represented by a set of five relations: relations between the model and its inputs, outputs, integrity constraints, validity evaluation, and computational subroutines, as follows:

INPUT (Modelname, Inputs)
OUTPUT (Modelname, Outputs)
OPERATION (Modelname, Functions)
INTEGRITY (Modelname, Constraints)
VALIDITY (Modelname, Evaluation)

Each relation in the scheme represents a unique characteristic of a model. They should be read as "the inputs of <modelname> include <input1, input2,..>," "the outputs of <modelname> include <output1, output2,..>," and so forth. The first four relations are important to the formulation of a model graph and the fifth (validity relation) is important to the selection of models.

The advantages of this scheme are two-fold: first, it is non-procedural, i.e., the model builder specifies what the model is rather than how the model computes data. Second, it can be implemented easily in a symbolic language, such as PROLOG.

Corresponding to the graphical representation, the input and output relations are nodes for formulating a model graph and the operation relation represents an edge. Hence, a basic model identified by a unique name is a combination of one operation relation (an edge) and its associated input relation and output relation (two nodes). Different model names stand for different models even when they have the same input and output data attributes because they may use different computational functions to convert data.

The integrity relation checks integrity constraints of a model to determine whether the conditions under which the model is applicable are present in the problem. For example, the least squares linear regression technique requires that the number of cases must be larger than the number of independent variables plus 2. Unless this constraint is satisfied, the sales forecasting model using the regression approach should not be considered in formulating a model graph.

Because the validity of a model can only be assessed after it has been implemented, the validity relation of a model indicates the historical validity of a model in a specific context. In other words, it represents a kind of subjective confidence in the model based on the previous experience in that specific context. This is an important criterion for model selection. For example, in the case of forecasting future sales, our experience indicates that the accuracy of the moving average technique is very good if more than 2 years' historical data are available: however, the technique is poor for identifying the turning point in a trend (Chambers, Mullick, and Smith, 1971). The validity relation of a model using the moving average approach must reflect this fact or, at least, inform the user to check this before using the model.

The operation relation specifies computational functions used in a model. It is part of the interface between the logical integration of models indicated in a model graph and physical execution of the selected model stored in the model base.

Figure 2 is a sample representation of the EOQ model. The model has the integrity constraint that both the holding cost and ordering cost must be a constant in the period. The validity relation indicates that if the integrity constraints are satisfied. The validity of the model is 0.8. Here, the number indicates a subjective evaluation on the degree that the model is appropriate for solving the problem. It should be measured based on a set of pre-specified criteria. Depending on individual implementation, "0.8" may mean excellent or very good.

The reason for quantifying validity is that it can be manipulated and calculated to facilitate model selection in a model graph. An optimizing algorithm for selecting the best model among all alternatives for the user, based on the

Figure 2. Representation of the EOQ Model

OUTPUT(EOQModel, [Economic order quantity])

INPUT(EQOModel, [Demand, Ordering cost, Holding cost])

OPERATION(EOQModel, [EOQ subroute])

INTEGRITY(EOQModel, [Constant(Ordering cost, Holding cost)])

VALIDITY(EOQModel, [0.8])

expected validities of available models, will be described later.

In order to determine the validity value, an evaluation function is certainly required. The evaluation function may evaluate the validities of different models according to the user's revealed preference or some pre-determined criteria. Further discussions on this issue can be found in Liang and Jones (1986).

FORMULATION OF A MODEL GRAPH

The major motivation for developing the graphical representation of models is to build consulting capabilities in MMSs, which can provide advice concerning effective utilization of existing models in the model base. In order to develop the consulting capabilities, we need a mechanism for formulating model graphs, the basis on which advice is generated, and for implementing strategies for model selection.

Formulation of a model graph involves extensive search in the database and the model base. Many heuristics have been developed for creating and traversing a search tree (see [Rich, 1983; Gondran and Minoux, 1984], or other books in graph theory or artificial intelligence for a review), including depth-first search, breadth-first search, and best-first search. The depth-first search and the best-first search are usually more efficient than the breadth-first search in finding a satisfactory model. In this section, a mechanism for formulating the model graph will be presented. It is based on the depth-first search strategy and compatible with the previously described graph-based scheme for model representation.

The basic idea of the depth-first search is to pick up an alternative arbitrarily at every node and work forward from that alternative. Other alternatives at the same level are completely ignored as long as there is any hope of reaching the destination using the original choice. If the original choice is proved impossible to lead to a solution, then go back one level to work on another alternative.

Suppose a user has made a query and the requested information is not directly available in the database, procedures for applying the depth-first search to formulate a model graph are as follows:

Step 1: Search OUTPUT relation in the model base to see whether there is a model that produces the output.

Step 2: If a model is found, go to step 3 to search INPUT relation in order to find the required inputs for executing the model. Otherwise, stop the searching process, report that no model is available in the model base, and then ask the user to develop a new model.

Step 3: Pick up an input, search the database for availability.

Step 4: If the input is available in the database, retrieve it and then go to step 3 for other inputs.

Step 5: If the input data is not in the database, search OUTPUT relation in the model base to see whether it can be produced by a model.

Step 6: If a model is found, search INPUT relation in the model base to determine its associated inputs, and then go to step 3.

Step 7: Otherwise, prompt the user for the input data.

Step 8: If it is provided by the user, pick up another input, and then go to step 3. Otherwise, give up the model, check OUTPUT relation to see whether there is another model.

Step 9: If another model is found, go to step 5. Otherwise, pick up another input and go to step 3.

Step 10: If all inputs of a model are available, check the integrity relation of the model.

Step 11: If the integrity constraint is satisfied, add the model to the model graph. Then, go to step 5 to check whether there is another model.

Step 12: Otherwise, give up the model and then go to step 5 to look for more models.

Figure 3 illustrates the process of formulating a model graph. The procedures of the best-first search are basically the same as the depth-first search, except that the former employs an evaluation function to evaluate the potential of all possible paths before further investigation and gives better paths higher priority, i.e., models with higher confidence factors will be examined earlier. There are certainly other possible approaches for building the model graph. They will not be discussed here, however, because they can be derived from the procedures described previously.

In this algorithm, if the operation that picks up an input of a model and searches for the availability of the specific input is considered a basic operation in the model base and represented as an edge, then the formulated model graph will be an alternate AND/OR tree.

<<Definition 12>> A tree

A tree, T, is a graph containing one or more nodes such that

1. there is a specially designed node called root,

2. the remaining nodes are partitioned into n (n>=0) disjoint sets T1,...,Tn where each of these set is also a tree. T1,...,Tn are called the subtrees of the root.

<<Definition 13>> An AND/OR tree

An AND/OR tree is a tree that includes both AND nodes and OR nodes.

<<Definition 14>> An alternate AND/OR tree

An alternate AND/OR tree is a tree in which the AND node and OR node appear at alternate levels. In other word, if nodes at level m are AND nodes, then the nodes at level m+1 must be OR nodes.

[Example] Figure 4 is an alternate AND/OR graph, which is the model graph formulated for providing advice about the EOQ and demand forecasting problem described in the first section.

Proposition 1: The model graph formulated in the above algorithm is an alternate AND/OR graph.

Proof: The algorithm employs two kinds of operations: one is picking up an input, the other is finding possible models that produce the input. If the former operation is performed on the node, then the node becomes true only if the operation has been successfully applied to all inputs (i.e., this node is an AND node). If the latter operation is performed, then the node becomes true if any model in the model base is available (i.e., this node is an OR node). Since these two kinds of operations are applied alternately in the propagation process of the model graph, the formulated graph must be an alternate AND/OR tree (see Figure 4).

STRATEGIES FOR MODEL SELECTION

Given a formulated model graph for producing the desired information, there are two ways in which useful advice can be generated. First, the MMS may show all possibilities indicated in the model graph to the user and then allow the user to make the selection. Second, the system automatically computes validities (or confidence factors) of various routes in the graph, chooses either a satisfactory route or the optimal route for producing the requested output, and then provides advice based on the selected path.

Since a model base usually contains many models, the combinatorial explosion sometimes may make it a little bit unrealistic to present all possible alternatives and force a system to adopt the second one. In implementing the second approach, there are two different strategies: satisficing and optimizing.

The optimizing strategy requires that an MMS formulate a model graph and then evaluate all paths in the graph in order to find the best alternative. The satisficing strategy, on the other

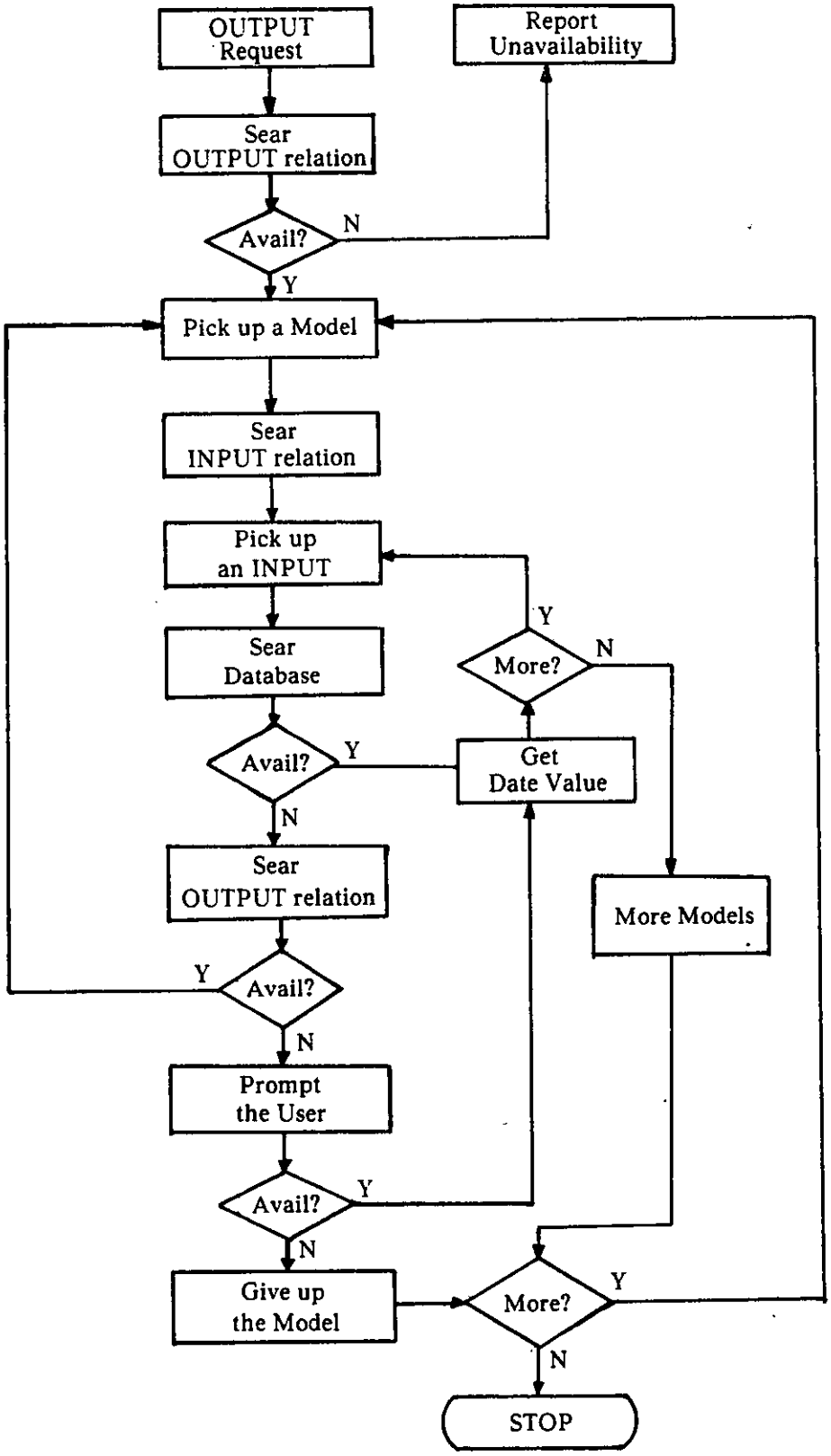


Figure 3. Process for Formulating a Model Graph

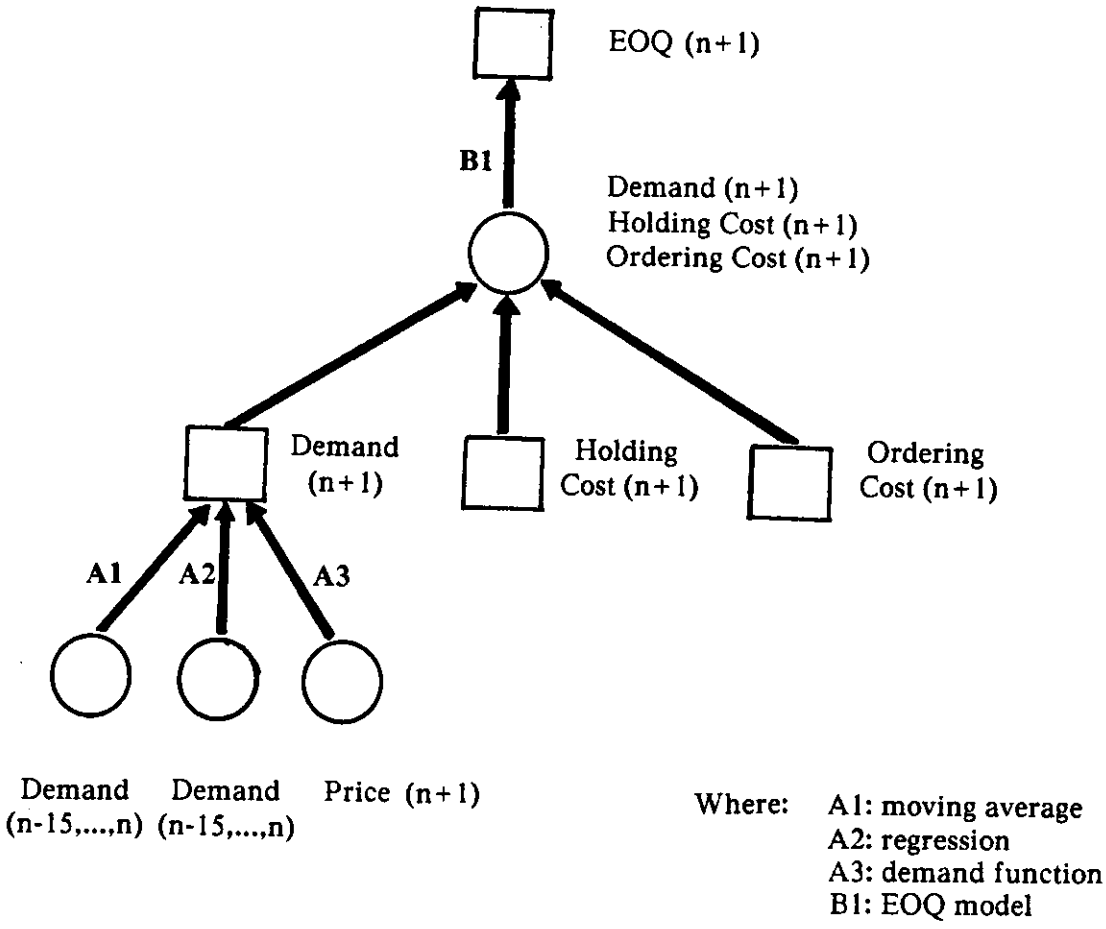


Figure 4. An Alternative AND/OR Graph

hand, requires that each path be evaluated immediately after it is found and accepted if it is satisfactory. Therefore, a complete model graph may not be required in the satisficing strategy.

If validities of all models in the model graph are available, then the optimizing strategy is simply to select the path with maximum validity. This can be formulated as a maximum validity flow problem, and most algorithms for finding the best path in graph theory can be applied to solve the problem. The objective is to maximize the validity of the selected route, subject to the constraints of modeling time, costs, and other considerations, as follows:

MAX validity

SUBJECT TO

1. Time constraint
2. Cost constraint

In order to reduce the complexity of the optimization problem, however, simple heuristics can be employed, as shown below:

Step 1: Determine the validity of each edge (a model) in the model graph. The system retrieves the validity of each selected member model by searching the VALIDITY relation of the model or executing the evaluation function if appropriate.

Step 2: Simplify the problem by removing dominated alternatives. If more than one edge is connecting two nodes, i.e., more than one model is available to convert an input to its associated output, then select the one with the highest validity and ignore the rest.

Step 3: Calculate validities for all possible paths from the initial state to the final state. Validity of a path is equal the product of the validities of its member edges.

Step 4: Select the path with the highest validity. The selection may be constrained by some other non-technical constraints, such as the computational cost, time and so forth. Therefore, it may also need an integer program, as described previously, to determine which path is the best one. However, because of the screening procedures described in steps 1-3, the new formulation should be more efficient than the original one.

The optimizing strategy guarantees that, given the criteria, the formulated model is the best available. Sometimes, however, the user may only need a satisfactory ad hoc model. In the satisficing strategy, the MMS follows the same procedure to formulate a model graph except that every path is evaluated at the time it is formulated. If a satisfactory path has been found, the process for formulating the model graph will be terminated. Figure 5 briefly illustrates the modeling process for the satisficing strategy.

IMPLEMENTATION OF THE FRAMEWORK

In order to demonstrate the feasibility of the graph-based framework for model management, a prototype, TIMMS (The Integrated Model Management System), has been implemented in PROLOG. The system supports the algorithm for formulating model graphs and a satisficing strategy that the system will provide advice based on the first alternative available. If the user does not like the first piece of advice and asks for more, the system will then provide the next alternative, if available, to the user. This process can go on until no more alternatives are available.

Figure 6 is a sample session of consultation for integrating the EOQ and demand forecasting models to produce "the EOQ for product a for 1987." The user specifies the desired information, the system first searches the database and finds that it is not available in the database. Then the system searches the model base, formulates a model graph as previously illustrated in Figure 4, and informs the user that the integration of the EOQ model and a demand forecasting model will be able to generate the desired information. The user may accept that advice and execute the integrated model, as shown in the session, or request more advice.

CONCLUDING REMARKS

The development of a model management system is an important but, as yet, poorly

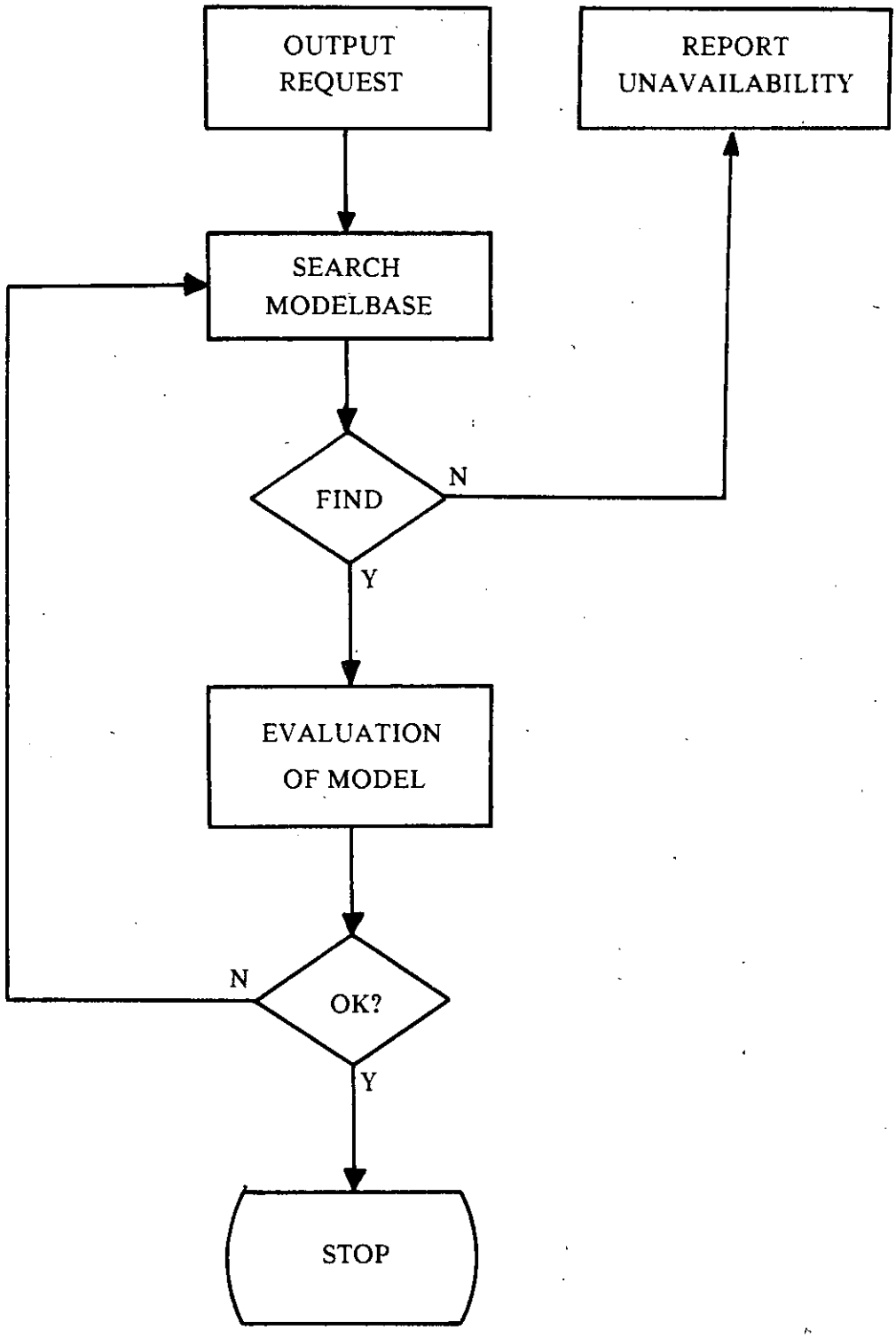


Figure 5. Process of the Satisficing Strategy

TIMMS: QUERY PROCESSING SUBSYSTEM)

Please specify the information you need:

OUTPUT: eoq
WHERE:
product = a
year = 1987

Please wait while checking the database
'eoq for a for 1987' is not available in the database

I am checking the model base

'demand for a for 1987' is needed but not available in the database
Could you provide it (y/n)? n

'price for a for 1987' is needed but not available in the database
Could you provide it (y/n)? y

Please enter the value: 10

MY SUGGESTIONS

There are three ways to produce the requested information

The first is:
Integrating model 'M1' and model 'M2'

Model 'M1' can generate 'eoq for a for 1987'

The execution of 'm1' needs the following 3 inputs:

- holding_cost of a
- ordering_cost of a
- demand for a for 1987

The database has ..... holding_cost of a = 5
the database has ..... ordering_cost of a = 20

'demand for a for 1987' can be produced by executing model 'M2'

The execution of 'M2' needs the following 1 input:

-- price of a

You provided ..... price of a = 10

Do you want to execute this model (y/n)? y

** eoq of a = 12

More suggestions (y/n)? n

THANK YOU

Figure 6. A Sample Session

researched area in information systems. Although some researchers have already studied the application of data models or knowledge representation schemes to MMS design, no mechanism of model integration and selection, which is essential to the generation of advice regarding effective use of models in the model base, has been developed.

The primary purpose of the paper was to develop mechanisms for creating capabilities of model integration and selection. In this article, a graph-based framework has been proposed. The graphical representation of models was defined. An algorithm for formulating a model graph was presented. And finally, strategies for model selection and an implementation of the framework were discussed.

REFERENCES

Blanning, R. W. "A Relational Framework for Model Management in Decision Support Systems," *DSS-82 Transactions,* 1982, pp. 16-28.

Blanning, R. W. "Issues in the Design of Relational Model Management Systems," *AFIPS Conference Proceedings,* 1983, pp. 395-401.

Blanning, R. W. "Conversing with Management Information Systems in Natural Language," *Communications of the ACM,* Volume 27, Number 3, 1984, pp. 201-207.

Blanning, R. W. "A Relational Framework for Joint Implementation in Model Management Systems," *Decision Support Systems,* Volume 1, Number 1, 1985, pp. 69-82.

Bonczek, R. H., Holsapple, C. W. and Whinston, A. B. "The Evolving Roles of Models in the Decision Support Systems," *Decision Sciences,* Volume 11, 1980, pp. 337-356.

Bonczek, R. H., Holsapple, C. W. and Whinston, A. B. "The Evolution from MIS to DSS: Extension of Data Management to Model Management," in *Decision Support Systems,* Ginzberg, *et.al.,* (eds.) North Holland Publishing, Amsterdam, Holland, 1981.

Bonczek, R. H., Holsapple, C. W. and Whinston, A. B. *Foundations of Decision Support Systems,* Academic Press, New York, New York, 1981.

Busacker, R. G. and Saaty, T. L. *Finite Graphs and Networks: An Introduction with Applications,* McGraw-Hill, New York, New York, 1965.

Carre, B. *Graphs and Networks,* Oxford University Press, Oxford, England, 1979.

Chambers, J. S., Mullick, S. K. and Smith, D. D. "How to Choose the Right Forecasting Techniques," *Harvard Business Review,* Volume 49, Number 4, 1971, pp. 55-64.

Dolk, D.R. *The Use of Abstractions in Model Management,* Ph.D. Dissertation, University of Arizona, 1982.

Dolk, D. R. and Konsynski, B. R. "Knowledge Representation for Model Management Systems," *IEEE Transactions on Software Engineering,* Volume SE-10, Number 6, 1984, pp. 619-628.

Donovan, J. J. "Database System Approach to Management Decision Support," *ACM Transactions on Database Systems,* Volume 1, Number 4, 1976, pp.344-369.

Elam, J. J. "Model Management Systems: A Framework for Development," *Proceedings of the 1980 SEAIDS,* 1980, pp. 35-38.

Elam, J. J., Henderson, J. C. and Miller, L.W. "Model Management Systems: An Approach to Decision Support in Complex Organizations," *Proceedings of the First International Conference on Information Systems,* 1980, pp. 98-110.

Geoffrion, A. M. *Structured Modeling,* Monograph, Graduate School of Management, University of California, Los Angeles, 1985.

Gondran, M. and Minoux, M. *Graphs and Algorithms,* translated by S. Vajda, John Wiley & Sons, Chichester, England, 1984.

Jones, C. V. *Graph-based Models,* Ph.D. Dissertation, Cornell University, 1984.

Konsynski, B. R. and Dolk, D. R. "Knowledge Abstractions in Model Management," *DSS-82 Transactions,* 1982, pp. 187-202.

Liang, T. P. "Integrating Model Management with Data Management in Decision Support Systems," *Decision Support Systems,* Volume 1, Number 3, 1985, pp. 221-232.

Liang, T. P. and Jones, C. V. "Meta-Design Considerations in Developing Model Management Systems," Working paper, The Wharton School, University of Pennsylvania, 1986.

Rich, E. *Artificial Intelligence,* McGraw-Hill, New York, New York, 1983.

Simon, H. A. *The Science of the Artificial,* The MIT Press, Cambridge, Massachusetts, 1981.

Sprague, R. H. and Carlson, E. D. *Building Effective Decision Support Systems,* Prentice-

Hall, Englewood Cliffs, New Jersey, 1982.
Sprague, R. H. and Watson, H. J. "Model Management in MIS," *Proceedings of the 7th National AIDS Meeting,* 1975, pp. 213-215.
Stohr, E. A. and Tanniru, M. R. "A Database for Operations Research Models," *International Journal of Policy Analysis and Information Systems,* Volume 4, Number 1, 1980, pp. 105-121.
Watson, G. W. *Knowledge Base Management for Model Management,* Master's Thesis, Naval Post Graduate School, 1983.
Will, H. J. "Model Management Systems," in *Information Systems and Organization Structure,* Grochla and Szyperski (eds.), Walter de Gruyter, Berlin, Germany, 1975, pp. 467-482.