Information Processing & Management Vol. 25, No. 6, pp. 629–646, 1989
Printed in Great Britain.

0306-4573/89 $3.00 + .00

NOVEL APPLICATIONS OF INFORMATION RETRIEVAL TO THE STORAGE AND MANAGEMENT OF COMPUTER MODELS

MICHAEL D. GORDON* and JAMES P. FRY
Computer and Information Systems, Graduate School of Business, University of Michigan, Ann Arbor, MI 48109-1234, U.S.A.

(*Received* 29 *August* 1988; *accepted in final form* 15 *March* 1989)

Abstract—Decision-making is often supported by computer-based models. To become a truly valuable corporate resource, such models must be easy to locate, share, and reuse. We describe a technical approach to model management aimed at establishing a database of computer models which can either be reused without modification or modified and composed with other models to assist with novel decisions.

We describe a formalism for representing models and discuss various types of queries supported by the formalism. We discuss important research issues that must be addressed for successful application of the formalism to model management. Further, we argue that the field of information retrieval is the natural referent discipline for the study of model storage and retrieval and advance the argument that retrieving computer-based models has an important connection to text retrieval based on documents' structure.

1. INTRODUCTION

Decision making in business and government is often supported by computer-based models which offer convenient abstractions of the real world. However, to make such models a truly valuable corporate resource, two efforts must be made: To assure that they are conceptually valid and free from programming errors (i.e., verified), and to share these valid, verified models among decision makers with similar needs in order to contain development costs and avoid the penalties of basing decisions on flawed models. This paper addresses the second of these efforts.

Computer models range in complexity from fairly simple "spread-sheet" models to very complex models programmed in languages such as Fortran and GPSS. Properly used, computer-based models provide enormous benefits for managerial decision making. For example, General Motors/Electronic Data Systems has documented over $1 billion in cost savings from the use of their multi-national planning system, PLANETS [1]. However, the creation of useful models is a complex task and the literature is replete with examples of multi-million dollar mistakes in planning and decision-making arising from "bad" computer models [2].

Like other software, computer models follow a life cycle. Useful models require substantial effort from requirements analysis through coding, testing, and maintenance. Complex models are expensive to build, hard for the non-developer to understand, and often only used once [3]. Even in an end-user modeling environment, not sharing models can result in significant duplication of effort by the constant re-invention of the wheel. Worse, errors invade models which are built by those without the patience or aptitude to test them thoroughly. Conceptual mistakes are introduced by failing to capture properly the real-world phenomena being modeled. For instance, models may oversimplify by omitting certain influences or their interactions or rest on assumptions (perhaps about interest rates) that are not borne out. "Programming" errors, such as data entry errors and subtler mistakes in formulas or statements, can further reduce the confidence one places in a model.

*Author received support for this research from the Graduate School of Business, Division of Research, The University of Michigan.

In fact, estimates suggest that at least one model in three contains some kind of error. Nonetheless, there is an increasing use of models for decision support at all organizational levels [4].

We describe a technical approach to the problem of model sharing based on the results of information retrieval research and theory. We describe a formalism for model description that holds particular promise for model sharing, and we discuss efforts to extend it by artificial intelligence methods. As a result, powerful methods are presented for representing and locating models, including methods exploiting their semantic structure and methods permitting graphical searching. We discuss both the benefits of these methods as well as important research questions which must be addressed to exploit them fully.

Importantly, this paper argues that the field of information retrieval is the natural referent discipline for the study of model storage and retrieval. We discuss both difficulties that have challenged information retrieval as well as principles and techniques for combatting them. Thus, we hope to provide guidance for others who are attempting to design systems for storing and retrieving models (or model sub-components). In addition, we feel, our efforts will help answer an important question for information retrieval, namely: How effective can "structured" information retrieval be?

2. AN INFORMATION RETRIEVAL APPROACH TO MODEL SHARING

We envision the day when a member of an organization can retrieve from a large "model base" a pre-defined and tested model that perfectly represents the situation he or she needs to study. More ambitious still, a modeler should be able to retrieve and splice together parts of several existing models to obtain a ready-to-use, verified, and validated model which applies to a novel modeling need [5]. This is not so far fetched as it may seem, Japan's Ministry for International Trade and Industry has sponsored a major national project, Sigma, to create a national software library from which Japanese companies can obtain free subroutines together with guidelines telling how these can be combined with other subroutines to create usable programs [6]. As Sprague and Carlson point out, "a comprehensive set of integrated models for decision support becomes a major corporate resource just as the database" [3].

To provide the ability to share and re-use models, it is necessary to provide means for cataloging, storing, and retrieving them. The field of information retrieval addresses these issues. The three fundamental processes in information retrieval are: querying, i.e., specifying a searcher's information need with a machine processable query; indexing, i.e., describing the subject content of documents in a format which the computer can process for making retrieval decisions; and matching the searcher's query and the indexed representations of documents to determine which documents to present to the searcher.

Information retrieval can involve objects of various types—documents, images, photographs, or computer models. In practice, models are catalogued and stored in libraries of computer subroutines or modeling source statements. Such storage allows for keyword-based access supported by file systems working with the computer operating system. Suggestions in the scientific literature have also been made for more sophisticated methods for storage and retrieval of models.

To date, no studies have been conducted which measure the retrieval effectiveness of model-based retrieval. But, effective information retrieval is known to be a difficult, sometimes deceiving, problem [7].

However, there is a "structure" embedded in computer models that presents special opportunity for using advanced retrieval methods with a high likelihood of success. This is a very attractive consideration in view of the growing need to retrieve relevant models. This paper will explore how structure leads to advanced retrieval. First, however, we examine other research suggestions for retrieving stored computer models, and other methods for using "structure" to assist in retrieving documents.

3. RELATED RESEARCH

In this section we consider the previous work that has been done in model management (storage and retrieval) as well as research directed at using the structure of text to

improve retrieval. We also comment on the similarities between model retrieval and software reuse.

3.1 *Model management: knowledge representation and other approaches*

Computer science (artificial intelligence) provides four widely accepted knowledge representation schemes [8,9]: production systems, semantic networks, frames, and first order logic. These knowledge representation schemes, in addition to relational database management methods, serve as a basis for classifying the approaches to model management research [10].

Elam, Henderson and Miller use semantic networks to represent four "nets" of knowledge [11]. A technical net represents the technical information about each model stored. A model net contains the objective function values for a specific model derived from network optimization routines. A language net contains user defined labels delineating access to common concepts. And an application net stores problem-specific information. Model selection is accomplished through an interrogation component which selects, classifies, and builds models. One advantage of the inheritance property of semantic networks is that the relationship among the concepts is preserved, making it relatively easy to categorize and classify knowledge. One major limitation is that it is difficult to represent a wide range of conditions found in a complex problem.

Bonczek, Holsapple, and Winston have proposed a model management architecture in which application-specific modeling knowledge is represented in the form of axioms, clauses, and well-formed expressions [12]. Reasoning is carried out using the predicate calculus. The major advantage of this approach is that it employs a well-researched and powerful reasoning process, and many successful examples of artificial intelligence systems use this technology. The major disadvantage is that the use of formal logic usually results in a large search space of rules unless a powerful organizing framework is used. Another drawback is that formal logic often results in the loss of relationships among concepts by focusing on independent facts.

Konsynski and Dolk propose a representation for model abstraction which closely resembles a frame-based knowledge representation [13]. Such abstraction is analogous to data abstraction in programming languages, involving data objects, procedures, and assertions expressed in the first-order calculus. Data objects and types are used to describe model variables, and assertions specify information about—and the relationships among—data objects and procedures (functions). Data items, data types, and procedures are assumed to be predicates, while assertions are well-formed functions. The advantage of this approach is that it combines the positive aspects of formal logic and semantic nets. The major disadvantage is that the frames must be narrowly pre-defined for the context of the problems in which they are needed.

Blanning's work uses the representation capabilities of the relational data model [14]. The decision model is represented as a set of relations which include the input and output criteria for the model. The "rows" of the relation do not exist in stored form. Rather, they are generated dynamically on demand, based upon the input and output attributes which are required to answer an interrogation. Model manipulation is performed using an extended query language similar to SQL which relies mainly on the selection function. (Blanning claims the join and projection operations are not needed because model instances are not stored within the system.) Models are instantiated by a "join process" of submodels in which the order of calculation directs the output of one submodel to be the input of the next submodel. A major advantage of this approach is its ability to couple the data storage component with the model representation component. In addition, it utilizes the full power of the relational model and its processing languages. A disadvantage of this approach is that the relationship between the problem and model are difficult to describe completely, since tabular representations are not well suited to problem descriptions.

Another recent effort in model management research, structured modeling, follows the data modeling paradigm and borrows from the software engineering discipline. Geoffrion's structured modeling provides a formal mathematical framework for representing a wide range of computer based models [15]. This framework is hierarchically organized and represented in the form of an acyclic graph which implements model semantics

and represents its mathematical structure. Geoffrion's work is quite broad, serving to "integrate" various paradigms: MS/OR, database management, programming languages, and artificial intelligence. As a consequence, it is difficult to master and not natural to the modeler.

As we have mentioned, none of these approaches has led to an implemented model base on which retrieval experiments have been performed. Thus, based on empirical results, the question of how models should be stored and retrieved is completely open. Since the benefits from solving this difficult question are so great, the accumulation of results, methods, and insights about storage and retrieval from the field of information retrieval ought to be exploited in making informed decisions about model retrieval. We begin to explore this referent discipline by next examining previous work concerned with two aspects of "structure": the structure of concepts within a given text, and the semantic structure of a literature as a whole.

3.2 *Structure in text retrieval*

Recent research suggests that the performance of information retrieval systems can be improved by taking advantage of structured information. As a highly stylized example, a history document may be structured with three fields representing, respectively, who conquered, whom and when. So structured, people in need of relevant history, documents may improve their chances of retrieving them by being able to search using any of these fields. In general, a structured document representation improves the chances for successful retrieval by allowing the searcher to ask more specific, more knowledgeable queries by exploiting a known organizational structure of documents. We now describe actual information retrieval research performed to take advantage of structure.

Croft and Lewis [16] are developing case frame representations for expressing searchers' information needs as well as for representing documents. Their effort is specialized to the domain of science and technology, for which they have established appropriate structures for representation. "Understanding" of documents is guided by structured models of searchers' requests.

Chi *et al.* [17] have developed structured representations of doctors' progress notes. Instances of the resulting structures are of a certain type: medical treatment, non-medical treatment, patient state, etc. With these methods, it is hoped that better medical care can be rendered by permitting better retrieval of this medical information, especially for patients whose records have become quite extensive over time.

Somerville and Wood [18] have argued that software reuse reduces development costs and testing costs (sentiments we share with respect to reusing model components rather than always building models from scratch). As a step in this direction, these authors capture knowledge about specific $UNIX_{TM}$ commands in structures they refer to as "software function frames." Like artificial intelligence frames, these structures specify templates ready to be filled in. (For instance, the software function frame for the action "search" designates that specific searching commands are differentiated based on their name, where they search, and what they search for. The command "grep" searches files for patterns, for example.)

The RUBRIC system [19,20] operates on the full text of a document to determine whether it should be presented to a specific user who repeatedly queries the system with known, relatively stable, needs for political information. It uses rules designed by that user. Some rules help define highly specific concepts the user is interested in, and others help indicate how specific words in document texts should serve as evidence for the occurrence of a topic. The rules rely on there being consistent structure in the domain of political events under consideration.

A domain independent system is Malone et al.'s [21] LENS system, which serves as a communication medium in addition to providing a strict information retrieval capability. This prototype system allows various forms of "semi-structured" messages to be created and sent. Although, in principle, there can be any number of different kinds of semi-structured messages, it is only by agreeing to use common structures that sophisticated message sending and retrieval is facilitated. For instance, LENS' production rule (IF . . .

THEN . . .) component allows messages to be routed, categorized, and assigned various priorities based on structure.

Each of the above systems is built upon the principle that there is structure to a particular field of study (science and technology, medicine, software functions, politics) that can be examined, modeled, and exploited for better retrieval. We will argue (section 4) that the same principle holds even more strongly with respect to computer models for business decision making.

Next, we consider some examples of classification structures such as thesauruses, dictionaries, and semantic "road maps." Each of these techniques suggests that establishing interrelationships among concepts offers opportunity for improved retrieval.

The MeSH classification structure is an important organizational technique used by the National Library of Medicine's online medical information retrieval system, MEDLINE. MeSH implements an 8-level "tree" covering approximately 14,000 terms [22]. It is used by manual indexers in conjunction with a set of rules to describe the medical periodical literature and also supports the formation and expression of queries by trained and untrained searchers. Using an algorithm to merge the MeSH thesaurus with the ACM Computing Reviews classification, Rada confirmed the usefulness of a merged thesaurus in improving retrieval by improving the conceptual organization of information [23].

Humphrey [24], also working at the National Library of Medicine, is conducting an investigation directed at knowledge-based indexing to improve the consistency in keyword phrases that are applied to documents. The system suggests, restricts, and automatically supplies certain index phrases based on other phrases applied already. The system makes use of inherited knowledge.

A more general view of classification was presented by Doyle [25], who argued long ago that associations among subject terms could be profitably exploited in retrieval. At that time, powerful computers and software were not available to test such ideas. Today we see ambitious projects that are taking advantage of current technology, such as Fox's efforts to build an enormous semantic net out of several online dictionaries [26,27].

3.3 *Software reuse*

Software reuse provides another perspective from which to examine the problem of model retrieval. We consider here the task of locating relevant software components (program designs, coded functions or procedures, entire programs, etc.). We ignore the other important software reuse tasks of comprehending, modifying, and composing software components.

Burton *et al.* [28] establish a database of reuse components (segments of code mostly written in Ada). To permit their retrieval, the following material is stored in a database: the machine running the code, the language in which the code is written, its documentation, hierarchically arranged category codes describing functionality, and five or fewer user-assigned keywords. Queries allowing specification of the type of component needed (e.g., components involving "stacks") and the relative importance of various aspects of components (code length, code readability, etc.) are input into a "scoring" algorithm which returns to searchers a set of components ordered by their predicted usefulness.

Prieto-Diaz and Freeman [29] search a library of reuse components classified on several dimensions including: component function, type of software in which the component would be used (e.g. compiler), functional area to which the component applies (e.g. CAD or auditing). The matching function ranks components by estimating how much effort would be required to reuse them for a particular query that supplies values for the same dimensions.

Other software retrieval methods resemble those used for model retrieval. The Paris system [30] uses theorem proving to derive a list of candidate components from their properties. A frame-based system revealing both default and exceptional values for different classes of code assists with software design, construction, and maintenance [31].

For two decades, the idea that new programs ought to take advantage of existing ones has excited software developers. Though newer programming techniques, including object oriented programming and data type abstraction, make it easier to design and reuse soft-

ware components, concomitant increases in the ability to locate code for reuse have not occurred. Biggerstaff and Richter [32] suggest that, for software reuse to deliver on its promise, breakthroughs in representation must allow the clear expression of what a software component does, how it does it, and how it is to be connected to other software components.

Similarly, we feel that the appropriate theoretical basis for model retrieval begins with an appropriate representation: a model skeleton (i.e., an augmented "influence diagram"—which Howard [33] has described as the "greatest advance . . . in the communication, elicitation, and detailed representation of knowledge"). We speculate that advances in model retrieval can shed light on software reuse and vice versa. At present, the means for selecting relevant software components from a software inventory remains a difficult problem for software reuse.

4. MODEL RETRIEVAL BASED ON "STRUCTURED" INFORMATION RETRIEVAL

The three fundamental processes of information retrieval provide a foundation for sharing and reusing computer models. We first discuss the indexing of computer models by means of a formal representation of their structure. We show, too, how a "concept base" can augment this representation by accounting for semantic difficulties lying outside the formalism. We next discuss querying for stored models in conjunction with the formal model representation. Finally, we describe important matching questions demanding additional research.

4.1 *Formal theory for the representation of models*

The results of Spiguel suggest that deterministic, discrete-time models can be precisely represented by a mathematical formalism based on set-theory [34]. Specifically, by describing the following sets, any model of this important class can be described completely and unambiguously:

1. The set of variables employed in the model;
2. The set of structural relationships that exist among variables (these designate which model variables influence the values of which other model variables);
3. The set of temporal relationships included in a programmed model (these designate how the values of model variables interact with each other over time); and
4. The set of mathematical equations which specify precisely how model variables interact.

For example, a very small programmed model which describes activity for a savings account might be expressed in a computer modeling language (pseudo-IFPS) as follows:

MODEL

```
Years 1988-1989
Balance = 50, Previous Total
Deposit = 10
Total = Compounds (Balance + Deposit, 10%)
```

When executed, this non-procedural language would produce output similar to:

OUTPUT

	1988	1989
Balance	50	66
Deposit	10	10
Total	66	83.6

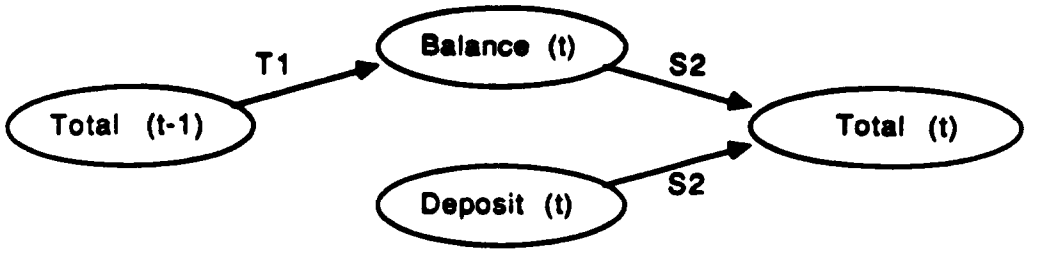


Fig. 1. Skeleton of programmed model.

This model shows what happens when, on January 1, 1988, and January 1, 1989, a depositor adds $10 to a savings account with a closing balance of $50 on the last day of 1987, at a bank which gives 10% interest each December 31st to any deposit in the bank for the last 365 days.

Using Spiguel's formalism, a skeleton of this programmed model is shown in Fig. 1. This skeleton shows:

- Model variables are the set {Balance, Deposit, Total}.
- The structural relationship S_2 involves the joint influence of variables Balance and Deposit on the value of Total.
- The beginning Balance in a model year is equal to the value of the Total in the PREVIOUS year, as reflected by the temporal relationship T_1.

The formal specification of the model also includes the fact that:

- The mathematical relationship between Total and the variables Balance and Deposit involves the functional relationship Compounds.

It has been proven that a deterministic, discrete-time programmed model, as expressed in some modeling source code, is isomorphic to its representation expressed by its skeleton and functional relationships [34]. That is, the formalism and the programmed model contain the exact same information. (In addition to the mathematical proofs documenting this equivalence, LISP encodings of programmed models expressed by the formalism have been used as the input to a compiler which produces IFPS code.)

This isomorphism between computer models and their formal representation means the formalism is an extremely powerful means for representing computer models for purposes of storage and retrieval. In essence, this equivalence means that a computer model is completely and perfectly represented by the formalism. Thus, in our approach, the set of model variables and the structural, temporal, and functional relationships associated with a model are encoded and stored in the model base to represent the model.

In this way, half the problem of retrieval by "structure" is solved—namely the problem of how to devise a structure which represents some object of retrieval (computer model or text) without loss of information. In contrast, previous efforts involving structured retrieval have been forced to choose a less-than-complete, and therefore somewhat arbitrary, means of representing a text.

4.2 *Semantic problems in representation: the "Concept Base"*

We must recognize that any formalism is built of primitive constructs which require "correspondence rules" to explain their application to the real world [35]. For instance, our formalism defines a representation for describing models in terms of sets of variables and sets of relationships. But it does not prescribe which models will be built, what their variable names will be, or what relationships will exist among these variables. In fact, retrieval will always be beset by linguistic and semantic problems that lie outside any formalism. We make provision for these difficulties by augmenting the formalism for model representation with a "concept base."

One of the problems lying outside our formal treatment is naming inconsistency. That

is, it is impossible to know the vocabulary that will be used to describe a model that performs a given task. For instance, the term "income" can be a synonym for "profit." But, in other contexts, its meaning is closer to "revenues." This is problematic for a modeler trying to retrieve models using the variable "income." Similarly, a modeler wishing to retrieve models dealing with "assets" will find the formalism incapable of recognizing models involving "real assets" as possibly being relevant.

Swanson provides insightful examples further portraying the semantic difficulties that beset those who wish to retrieve stored information [36]:

1. Different people have very different opinions about whether a given document is relevant to a (supposedly) identical need for information.
2. There are seemingly countless subject terms that individuals will select to characterize a document's subject contents. Even so, they will rarely select the same terms used to describe it in an official card catalog.
3. Documents appropriately described by certain keywords often fail to use that keyword (or lexical variations) in their text.

To mitigate these retrieval problems, we look to enhance the formal representation of models. The work of Fox [26,27], Humphrey [24], and the foundational work of Doyle [25] motivate our construction of a "concept base" for describing the semantic relationships among standard business concepts. The concept base can account for both generalization and specialization among terms ("IS A" relationships) as well as part-constituent (aggregation/disaggregation) relationships. The concept base, similar to a thesaurus, allows for browsing of concepts in search of relevant models, enables "knowledge based indexing" which can suggest or preclude the use of certain combinations of keywords for describing models, and links specific model variables to standard business concepts to mitigate problems arising from inconsistent naming of model variables. An example demonstrates the use of the concept base to augment indexing.

Example 1. Model Entry into the Model Base. A modeler builds a small model to help her account for her expenses during the last year. Figure 2 shows a model skeleton representing the basic model. After model validation and testing, the model and its formal representation are entered into the model base. The concept base is used to provide additional indexing support:

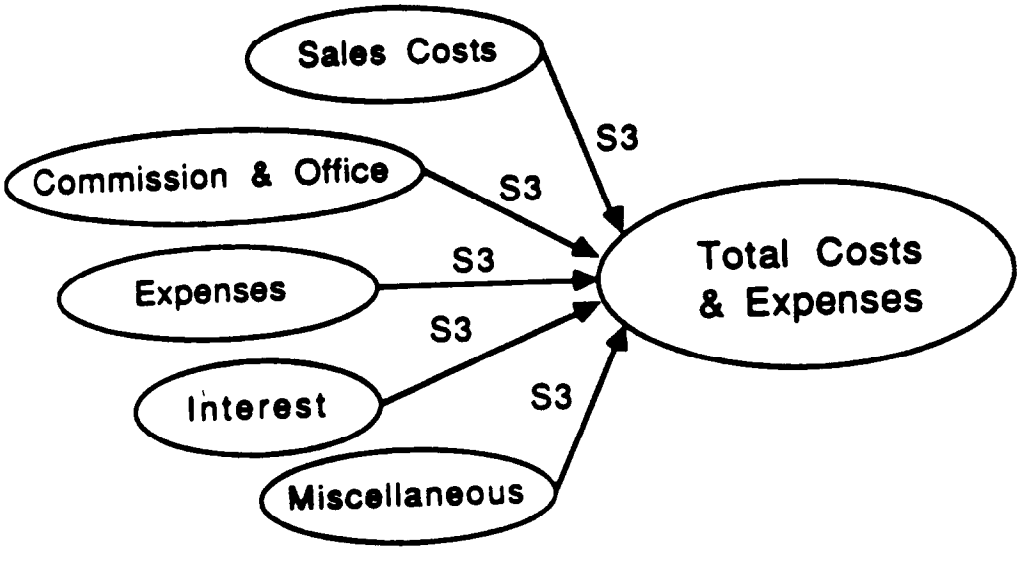


Fig. 2. The "Expenses" model skeleton.

1. The "concept base" is graphically presented to the user so she can catalog her work (see Fig. 3). By using a mouse, she traverses this structure, encountering "Financial Concepts," then "Income Statement," then "Expenses," then some specific expenses such as "Cost of Sales." Each of these terms is a "controlled"—i.e. official—vocabulary term used to help provide naming consistency.
2. Database links are established between the modeler's chosen variable names and the official vocabulary maintained in the "concept base." For instance, what the modeler calls "sales costs" the system calls "cost of sales," and what she calls "commission and office," the system calls "selling, and general administrative expenses." Pointing and mouse-clicking establishes these associations. The database links established from standard vocabulary to model variables are one-to-many, meaning that all model variables (in any stored model) related to a standard vocabulary item are linked to that item.

Information retrieval research suggests that several representations of a document are better than one for increasing the probability of retrieving all relevant documents [37,38].

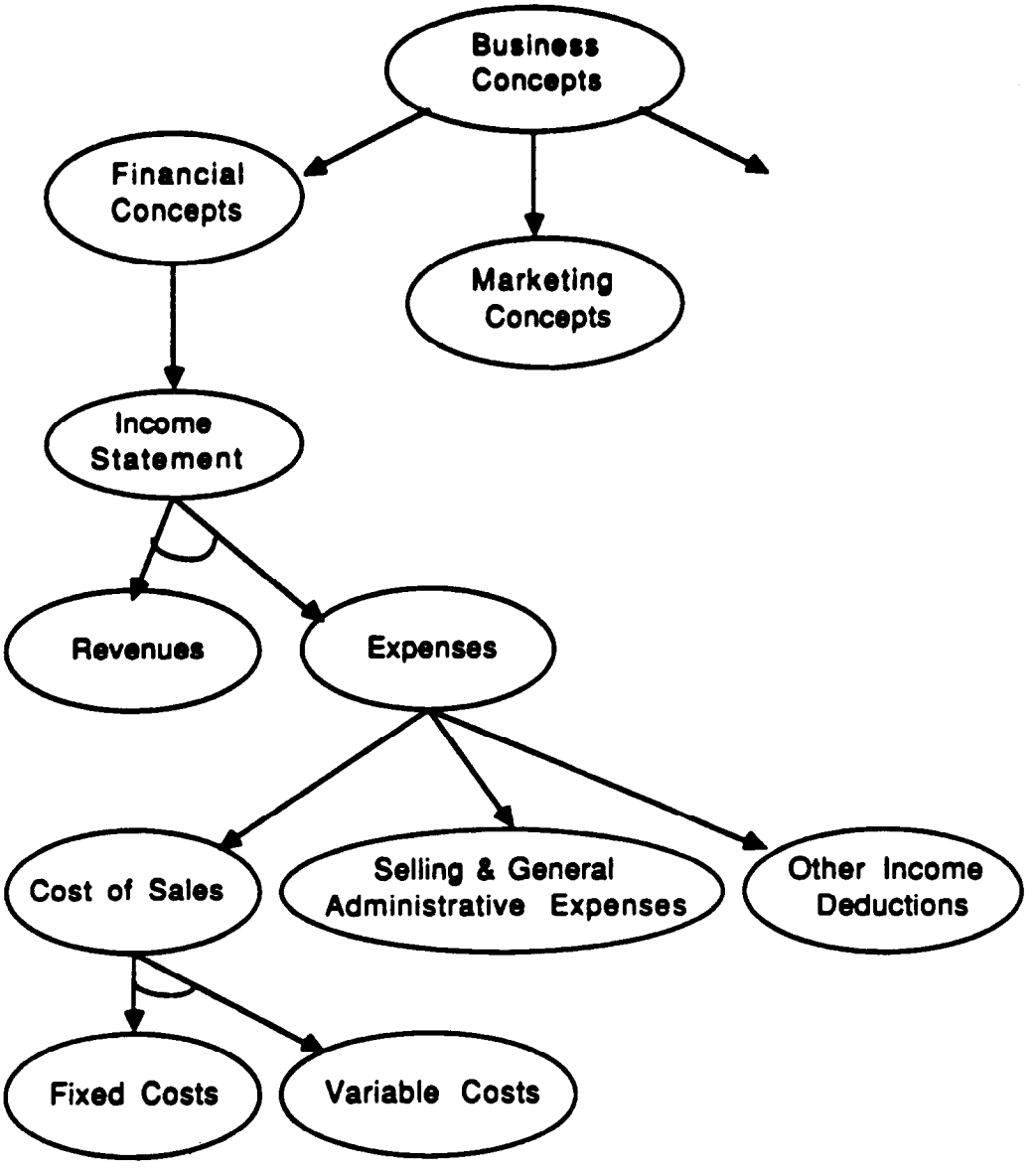


Fig. 3. Controlled vocabulary in Concept Base.

Since the cost of failing to retrieve a relevant model may be enormous due to either the effort required to rebuild it or to the consequences of basing a decision on a faulty model, it is extremely important that modelers retrieve all existing models of potential relevance. So, in addition to the indexing we have described, and using the concept base for assistance,

3. The model builder is prompted for a textual abstract.
4. The model builder is prompted for a keyword description of her model.

Note: This example shows the use of a concept base to help in naming consistency as well as the use of multiple representations (including keywords, abstracts, and labeling of modeling variables) to support improved retrieval by future model builders. We are performing experiments to test the effectiveness of each model representation separately as well as in combination.

4.3 *Model retrieval: querying*

The methods of model representation we have discussed, together with the semantic extensions offered by the concept base, provide several powerful methods of retrieval. We show by example how querying is performed using keyword-based and graphical methods.

Example 2: Retrieval from the Model Base by Concept: Keyword querying. A second model builder needs a model of an income statement. Though there is no such model in the model base, the system possesses some useful knowledge. It knows that the keyword description of the last model was "Expenses," (a controlled vocabulary term contained in the concept base), and that expenses and revenues are the two constituents of income statements. (Note the "AND" in Fig. 3 between "Expenses" and "Revenues" depicted by the arc connecting them.) Thus the system suggests that the modeler consider whether the last model can be of use.

Note: There are five potential ways the second modeler can use official vocabulary in the concept base to learn about the existence of the Expenses model component that he needs in building an income statement:

1. By choosing the controlled vocabulary term "Expenses" directly.
2. By choosing the controlled vocabulary term "Income statement." From this, he would be led to "Expenses," one of the two constituent parts of the Income Statement.
3. By choosing "Cost of Sales," an element under Expenses. The system would notify him about models involving Expenses in an attempt to generalize from a particular term to a broader modeling topic.
4. By performing a Boolean search on model abstracts, looking, for instance, for models about "Income statements" or "Revenues" or "Expenses."
5. By performing a Boolean search on the keyword descriptions maintained with models.

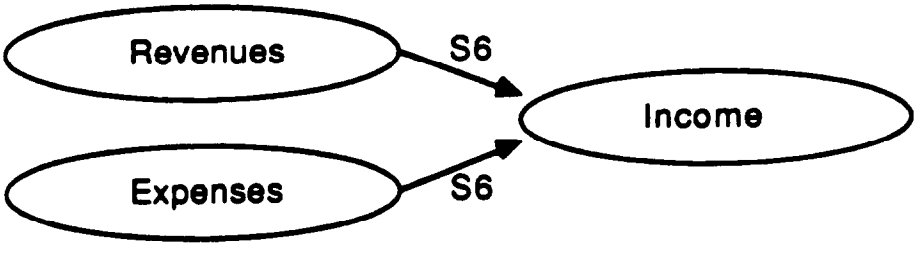


Fig. 4. The "Income Statement" model skeleton.

Once the modeler's Income Statement model is created (using the "Expenses" model as one of its sub-components; see Fig. 4), it will be catalogued in a fashion similar to the "Expenses" model. That is, its variables will be linked to official vocabulary in the concept base and it will receive a keyword description and an abstract. Also, its formal representation will be stored in the model base. Subsequent modelers will be able to retrieve it in its entirety, or retrieve just its "Revenues" or "Expenses" subcomponents.

This example has shown how the concept base also assists in retrieval. In general, it supports concept generalization, specialization, and aggregation/disaggregation (part-whole) relationships during retrieval. As Example 1 showed, the concept base can also help the model builder with indexing by offering her a "map" of concepts which she uses both for vocabulary control for model variables and for use in assigning keywords descriptions of models and in writing model abstracts. As Example 2 has suggested, the grain of retrieval can support location of complete, ready-to-use models or model sub-components which can be spliced together to form complete models.

The fact is that not only is there exploitable structure in the world of business, but model retrieval offers demonstrated opportunities for taking advantage of structure [1].

Example 3: Retrieval from the Model Base by Structure: Graphical Retrieval. Based on research suggesting that an improved graphical interface can enhance retrieval of documents [39,40,41,42], we allow the model retriever to use a mouse-with-windows interface to easily create graphical relationships that describe his or her modeling needs. For instance, using an interface similar to that used by the simulation tool STELLA [43], one can easily create the query below

$$(A) \rightarrow (B)$$

which shows an unspecified relationship between variables A and B.

We support this form of graphical representation of queries for three reasons. It is easy to use, supports the expression of complex modeling queries (involving relationships among variables), and is the most natural form of querying given the underlying stored representation of models.

In response to the query (A) → (B) the system will retrieve for the user models containing the two variables A and B with any kind of influence relationship between them. ("Influence" suggests there is a structural, temporal, or functional relationship.) Just as easily, the searcher can specify a particular relationship that must be present for retrieval. Thus, we provide such a facility by allowing searching based on:

1. changes to a variable over time,
2. standard business functions, or
3. other mathematical relationships.

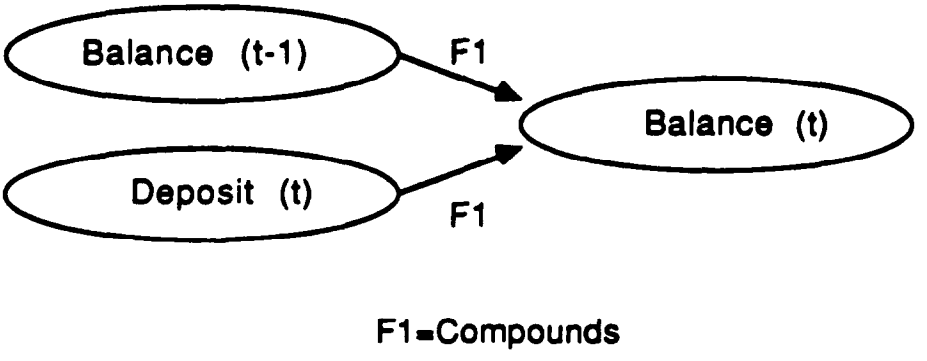


Fig. 5. Model skeleton fragment showing function "compounds".

Consider the model skeleton fragment in Fig. 5 which includes the functional relationship "compounds." This skeleton fragment indicates two things: a) The variable "balance" changes over time as a function of its previous value. (More generally, a variable may exhibit a temporal relationship to the previous value of any variable.) b) The model is concerned with the (presumed to be standard) function "compounds." Standard modeling functions we might wish to use for model retrieval purposes include: net present value, internal rate of return, etc. Functional relationships are also defined by arithmetic operators.

The temporal and functional information associated with this fragment would be stored in the Model Base. Their storage will permit modelers to retrieve the model when they issue queries of the following sort:

- "find models showing a change of balance over time" or
- "find models involving compounding."

These queries can be posed graphically or by use of keywords.

4.4 *Research on matching in structure-based retrieval*

The heart of our research concerns retrieval using graphical (structure-based) querying. Again suppose a user specifies her model retrieval need by (A) → (B). Even if there is no model with direct influence from (A) to (B), there may be models exhibiting influence close to this. For instance, (A) might influence (C), and (C) influence (B). These types of syntactic transitivities can be recognized following relational database theory [44]. "Conceptual" relationships are also very important. For instance, a good target for retrieval would be a model exhibiting

$$(A') \rightarrow (B)$$

where A′ is the "father" concept of B in the concept base.

We conclude this section, by describing some of the important issues concerning matching in supporting graphical, structure-based retrieval. We consider it essential to provide such retrieval support, and feel these issues are crucial to the success of "structured retrieval." We look to both information retrieval and artificial intelligence to help us solve these problems.

4.4.1 *Context.* Consider a modeler looking for models in which the variables "Income" and "Expenses" jointly influence the variable "Net Income." This is, suppose the modeler expresses his need for models as in Fig. 6. The problem of context is: Should any model which satisfies these influence requirements be automatically retrieved? Or, does the con-

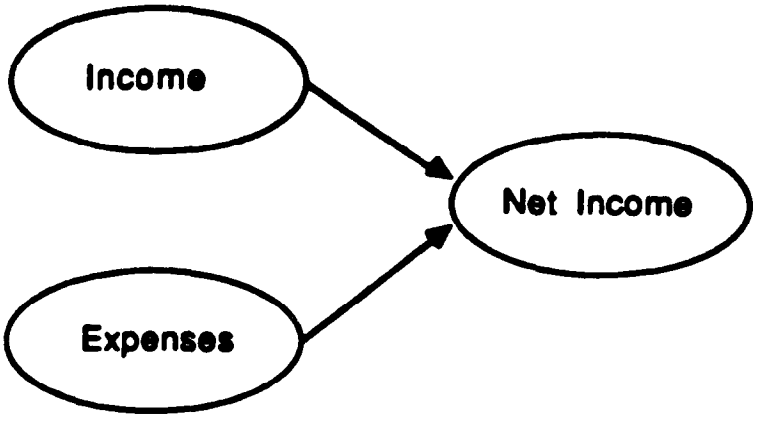
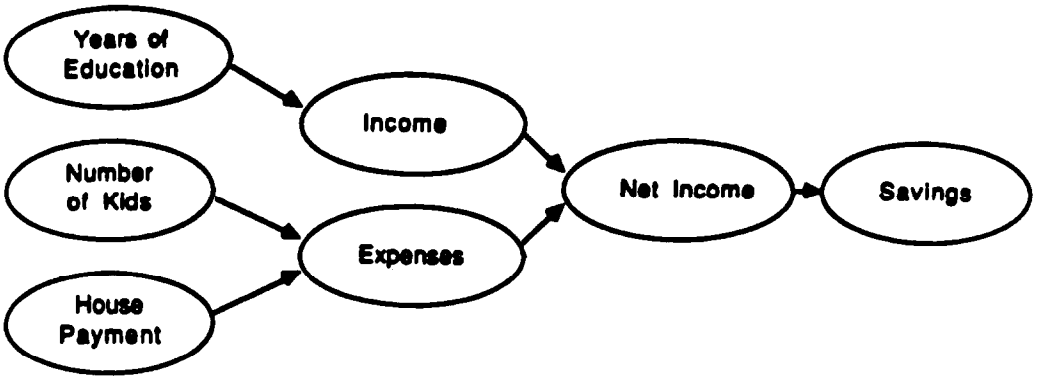
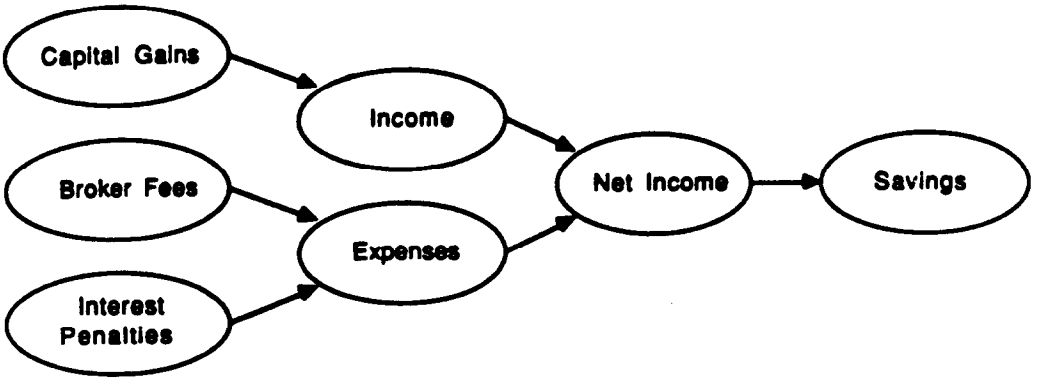


Fig. 6. Graphical query relating "income" "expenses" and "net income."



(a)



(b)

Fig. 7. Models involving "income," "expenses," and "net income."

text in which it is embedded in some way invalidate (or reduce the likelihood) of its relevance to the modeler?

Consider the models represented in Figs. 7a and 7b. (Here and elsewhere, arcs indicate structural, temporal, or functional relationships.) Though they both satisfy the modeler's specification, they seem to represent quite different modeling situations. The modeler, in specifying his need, will try to depict essential relationships. Context problems arise if there are ambiguous settings supporting these relationships, only some of which are appropriate to the modeler's need.

4.4.2 *Completeness.* Completeness is the problem of partial matching considered for structured retrieval. That is, when is there enough essential detail between the influence structure of a query and the influence structure of a model to constitute a "match," and when isn't there?

Consider a searcher in need of models in which stock prices are influenced by dividends, price-earnings ratios, and the stock's Beta (its volatility compared to the market as a whole). Figure 8 shows the skeleton of a model in which all the requisite variables and influences are present except the Beta (indicated by the fuzzy oval and arrow). It is plausible for the searcher to react in either of these ways to the model in the figure: "The model seems promising since dividends and the price-earnings ratio are much more important anyway." Or, "A stock's Beta is crucial in economic times like these. The model won't do."

In comparing a model to a query, either variables, or structural, temporal, or functional relationships may be missing. Most generally, the problem of completeness addresses how a machine can make guesses about model relevance or rank models by degree of similarity to a searcher's query.

4.4.3 *Naming.* Naming consistency poses considerable difficulties for retrieval of models or other objects. We have discussed already our attempts to control vocabulary by linking freely chosen model variables to official terms in the concept base. We have discussed,

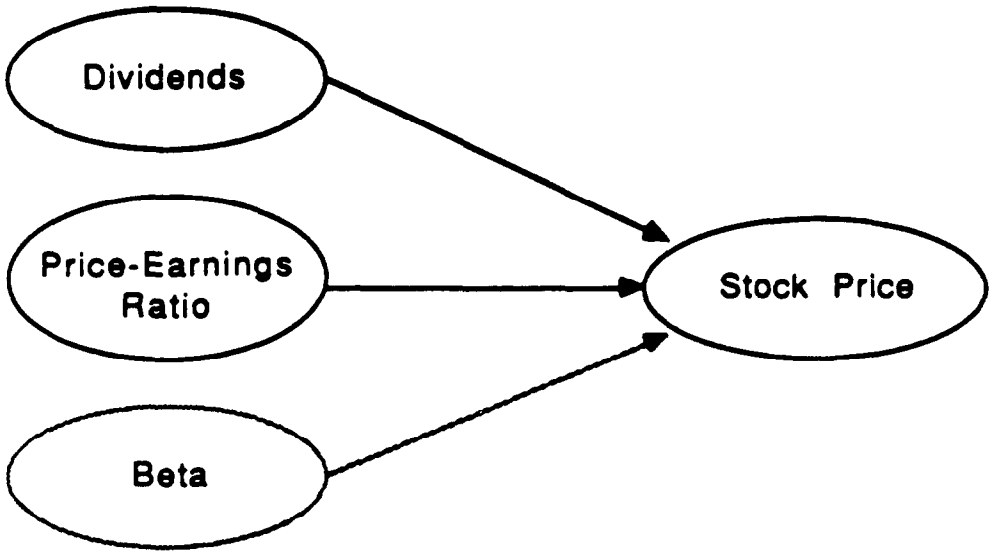


Fig. 8. Skeleton of a model partially matching a query.

too, how generalization, specialization, and aggregation/disaggregation are important mechanisms for considering differences between names used in a query and names associated with a model.

The most important naming problem is the degree of similarity among names. For instance (see Fig. 3), how similar are each of the following terms to the controlled term "Cost of Sales": Fixed Costs (a constituent of Cost of Sales); Expenses (a generalization); Other Income Deductions (a sibling)? Of course, other relationships within a network of concepts are possible, too, such as "grand child," "uncle," etc. Ultimately naming problems must be addressed by algorithms which select as relevant (or rank order) models in light of a query. It is not at all clear how algorithms can best perform this function in assessing the similarity between names used in queries and names used in model descriptions.

4.4.4 *Scale.* Problems of scale arise when there is a discrepancy between the detail included in a query and the detail included in a model. Consider the query depicted in Fig. 9a. It expresses a need for models in which inflation rate and amount of investment together have influence on internal rate of return. The influence structure of the model in Fig. 9b involves the three requisite variables, "Investment," "Inflation," and "IRR," and even has the appropriate (transitive) linkages. (For example, Inflation influences IRR through C.P.I. and Discount Rate.) Syntactically, such transitivities can be detected.

Semantically, however, one must consider whether (or to what degree) the query and the model skeleton stand for similar models. The model may, for instance, contain so much irrelevant detail that, in actuality, it has nothing to do with the modeler's needs.

Problems of scale involve:

1. decisions by the modeler concerning numbers of variables and amount of influence structure to include in a query;
2. decisions in representing a model about whether it should be represented in complete or partial detail; and
3. decisions in matching concerning similarity between model and query.

4.4.5 *Analogy.* Analogy offers great retrieval potential but more difficulty still in determining similarity between models and queries. Analogy seeks essential similarity in the face of apparent difference. It sacrifices surface similarity for deeper similarity.

Figure 10a represents a model which calculates the distance a plane flies as determined by its time in flight, its "wind-less speed," and the speed of the head wind into which it is flying. Figure 10b represents a drainage model, in which a pool has simultaneous inflow

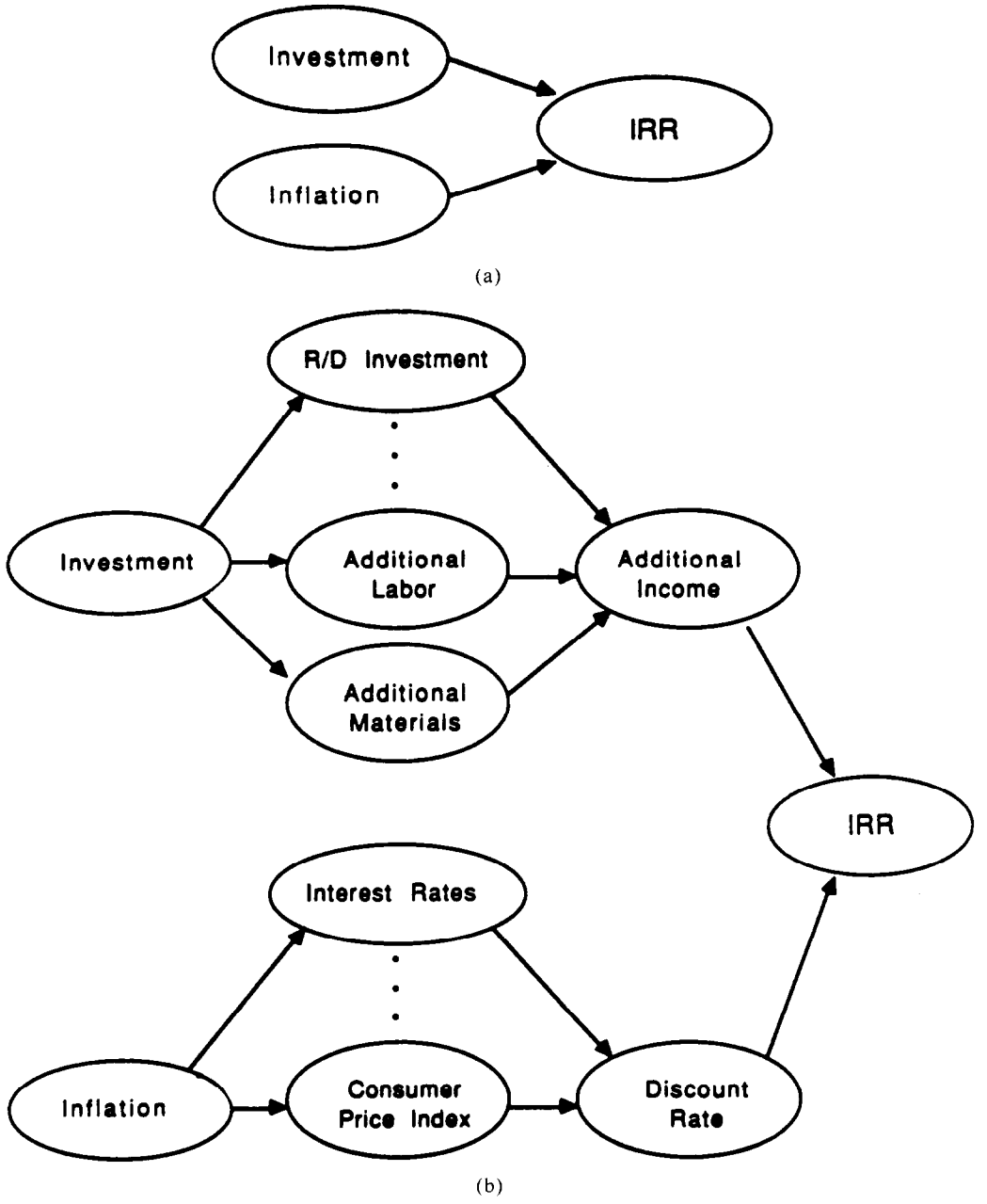


Fig. 9. Problems of scale in a model.

and outflow. In these models, "net rate" is analogous to "net speed," and "inflow rate" and "drainage" are analogous to "wind-less speed" and "wind speed," respectively. In principle, we can solve drainage problems with our airplane model.

The literature on analogy bears on four of the problems we have described: context, completeness, scale, and analogy (see, for example, [45,46,47]. The relevance of this literature stems from its concerns with distinguishing central and non-central components of a representation, developing transformations from one representation into another, and using learning together with analogy formation. We are exploring these ideas in our work.

5. CONCLUSION

Model sharing is necessary to contain the costs of model development and reduce the likelihood of basing decisions on faulty models. We have described a technical approach

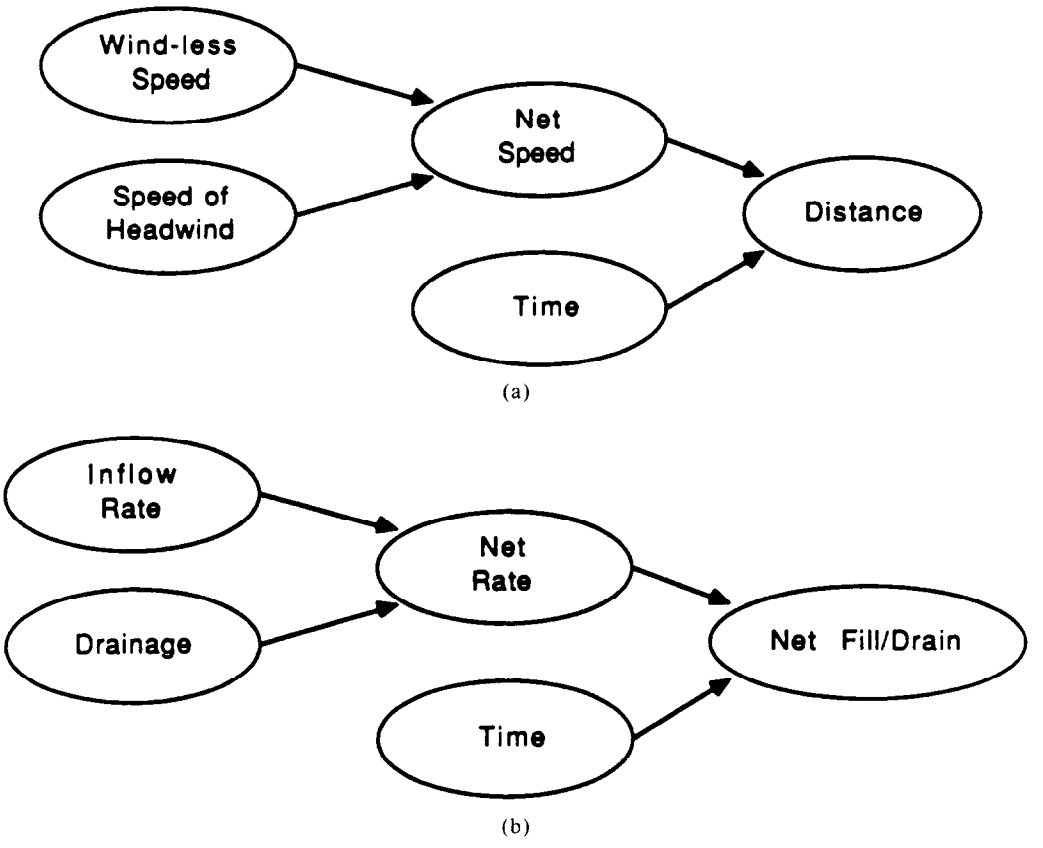


Fig. 10. Models demonstrating analogy.

to model sharing based on the idea of "structured" information retrieval. The structure we use for representing models is a set-theoretic formalism which provides a complete, accurate description of a model. We have discussed both the power of the formalism as well as the need to use semantic information to augment it.

Of course, the formalism for model representation cannot be used outside the domain of computer-models. But it provides an important context in which to address the second half of the "structured" retrieval problem. That is, given a representation which enables one to store complete and totally accurate information about all aspects of a model—its variables and all structural, temporal, and functional relationships among them—what is the most effective means of retrieval?

In fact, finding optimal retrieval methods which are based on this formalism is equivalent to answering: How effective can retrieval based on structure be? We are suggesting that "structured" retrieval depending on imperfect and incomplete methods of representing models or text can never be as effective in promoting retrieval as methods which are based on an isomorphic (perfect and complete) structured representation. And, if we believe that structured retrieval is inherently more powerful than retrieval not allowing structure (a hypothesis we plan to test, and as work on structured retrieval of text would suggest), answering the questions we have raised may give an indication of how effective retrieval may ultimately be. In this sense, we view this research as being as important for information retrieval as for model management.

A great deal of research needs to be done in order to determine the optimal methods for "structured" retrieval. Since a searcher's query rarely coincides completely with a document description (structured or not), matching algorithms for retrieving documents may allow for "partial matching," using various matching methods described in the literature (see, for example, [48] or [49]. These tend to be ad hoc and, for structured representations, quite specific to the type of structure being employed.

By developing an information retrieval-based approach to model storage and retrieval, we are attacking a problem of significant and increasing importance from the vantage of a strong referent discipline. Following this referent discipline, this paper has presented our technical approach to model indexing, querying, and matching. Important work remains to be done—work that can lead to both practical solutions for model retrieval as well as deep insights into structure-based information retrieval.

REFERENCES

1. Breitman, R.; Lucas, J. Planets: A modeling system for business planning. Interfaces 17(1): 94-106; 1987.
2. Business Week. How personal computers can trip up executives. 94-96; September 24, 1984.
3. Sprague, R.H.; Carlson, E.D. Building effective decision support systems. Englewood Cliffs, New Jersey: Prentice Hall; 1982.
4. Ulvila, J.W.; Brown, R.V. Decision Analysis Comes of Age. Harvard Business Review, Sep.-Oct.; 130-141; 1982.
5. Kottemann, J.; Dolk, D. Process-oriented model integration. Proceedings of the Twenty-First Annual Hawaii International Conference of Systems Sciences; 1988 Jan.; 396-402; Kona, Hawaii.
6. Haavind, R. Tools for compatibility. High Technology, 34-42; Aug. 1986.
7. Blair, D.; Maron, M.E. An evaluation of retrieval effectiveness for a full-text document-retrieval system. Comm. ACM, 28(3): 289-299; 1985.
8. McCalla, G.; Cerone, N. Approaches to knowledge representation. IEEE Computer, 16(10): 12-18; 1983.
9. Dutta, A.; Basu, A. An artificial intelligence approach to model management in decision systems. IEEE Computer, 89-97; Sept. 1984.
10. Applegate, L.M.; Klein, G.; Konsynski, B.; Nunamaker, J.F. Model management systems: proposed model representations and future designs. Proceedings Fourth Conference on Information Systems; 1-15; 1985 Dec. 16-18; Indianapolis.
11. Elam, J.; Henderson, J.; Miller, L. Model management systems: an approach to decision support in complex organizations. Proceedings First Conference on Information Systems; 98-110; Dec. 8-10, 1980; Philadelphia, PA.
12. Bonczek, H.; Hosapple, C.W.; Winston, A. Evolving roles of models in decision support systems. Decision Sciences, (11)2: 337-356; 1981.
13. Konsynski, B.; Dolk, D. Knowledge abstractions in model management. DSS-82 Transactions; 187-292; 1982.
14. Blanning, R.W. Issues in the design of relational model management systems. Proc. 1983 National Computer Conference; 395-401; 1983 May 16-19; Anaheim, CA.
15. Geoffrion, A. An introduction to structured modeling. Management Science, 33(5): 547-587; 1987.
16. Croft, W.B.; Lewis, D.D. An approach to natural language processing for document retrieval. Proceedings of the Tenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 1987 June, 26-31; New Orleans.
17. Chi, E.C.; Friedman, C.; Sager, N.; Lyman, M.S. Processing free-text input to obtain a database of medical information. Proceedings of the Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 82-90; June 5-7, 1985; Montreal.
18. Sommerville, I; Wood, M. A software components catalogue. In: Roy Davies, ed. Intelligent information systems: progress and prospects. Chichester, England: Ellis Horwood Limited; 1986; 13-32.
19. Tong, R.M.; Askman, V.N.; Cunningham, J.F.; Tollander, C.J. RUBRIC: an environment for full text information retrieval. Proceedings of the Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 243-251; 1985 June 5-7; Montreal.
20. Tong, R.M.; Appelbaum, L.A.; Askman, V.N.; Cunningham, J.F. Conceptual information retrieval using RUBRIC. Proceedings of the Tenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 1987 June, 247-253; New Orleans.
21. Malone, T.; Grant, K.R.; Turbak, F.A.; Brobst, S.A.; Cohen, M.D. Intelligent information-sharing systems. Comm. ACM, 30(5): 390-402; 1987.
22. Forsyth, R.; Rada, R. Machine learning: applications in expert systems and information retrieval. Chichester, England: Ellis Horwood Limited; 1986.
23. Rada, R. Knowledge-sparse and knowledge-rich learning in information retrieval. Information Processing & Management, 23(3): 195-210; 1987.
24. Humphrey, S. Illustrated description of an interactive knowledge-based indexing system. Proceedings of the Tenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 1987 June, 73-90; New Orleans, LA.
25. Doyle, L.B. Semantic road maps for literature searchers, Journal of the ACM, Vol. 8, 553-578; 1961.
26. Fox, E. Expert retrieval for users of computer based message systems. Proceedings of the Forty-Ninth Annual Meeting of ASIS; 88-95; Sept. 28-Oct. 2, 1986; Chicago, IL.
27. Fox, E. Response to: representation of dictionary information, IRList Digest, 3(18); July 9, 1987.
28. Burton, B.A.; Aragon, R.W.; Bailey, S.A.; Koehler, K.D.; Mayes, L.A. The reusable software library. IEEE Software, 25-33; July 1987.
29. Prieto-Diaz, R.; Freeman, P. Classifying software for reusability. IEEE Software, 6-16; July 1987.
30. Katz, S.; The, K. A preliminary report on the Paris system, Tech. Report STP-114-85, MCC, Austin, Texas, 1985.
31. Bassett, P.G. Frame-based software engineering. IEEE Software. 9-16; July 1987.
32. Biggerstaff, T.; Richter, C. Reusability framework, assessment, and directions. IEEE Software. 41-49; March 1987.
33. Howard, R.A. Decision analysis: practice and promise. Management Science, 34(6): 679-695; June 1988.

34. Spiguel, C.P. Computer aided modeling: an application to decision support in a business environment. Ph.D. Dissertation, The University of Michigan, Ann Arbor, Michigan, 1982.
35. Nagel, E. The structure of science. New York: Harcourt, Brace and World, 1961.
36. Swanson, D. Historical note: information retrieval and the future of an illusion. Journal of the American Society for Information Science, 39(2): 92–98; 1988.
37. Katzer, J.; McGill, M.J.; Tessier, J.A.; Frakes, W.; Das Gupta, P. A study of the overlap among document representations. Information Technology: Research and Development, 1(4): 261–274; 1982.
38. Tenopir, C. Full text database retrieval performance. Online Review, 9(2): 149–163; 1985.
39. Belkin, N.J., Oddy, R.N.; Brooks, H.M. ASK for information retrieval: Part I. Background and theory. Journal of Documentation, 38(2): 61–71; 1982.
40. Belkin, N.J.; Oddy, R.N.; Brooks, H.M. ASK for information retrieval: Part II. Results of a design study. Journal of Documentation, 38(3): 145–164; 1982.
41. Oddy, R.N. Information retrieval through man-machine dialogue. Journal of Documentation, 33(1): 1–14; 1977.
42. Frei, H.P.; Jauslin, J.F. Graphical presentation of information and services: A user-oriented interface. Information Technology, Vol. 2: 23–42; 1983.
43. Richmond, B. A user's guide to STELLA. Lyme, N.H.: High Performance Systems, Inc.; Nov. 1985.
44. Maier, D. The theory of relational databases. Rockville, Maryland: Computer Science Press, Inc.; 1983.
45. Hofstadter, D.R.; Mitchell, M.; French, R.M. Fluid concepts and creative analogies: A theory and its computer implementation. Cognitive Science and Machine Intelligence Laboratory, University of Michigan. Technical Report 10; 1987.
46. Carbonnel, J.G. Learning by analogy: formulating and generalizing plans from past experience. In: R.S. Michalski, J.G. Carbonnel, and T.M. Mitchell, eds. Machine learning: An artificial intelligence approach. Palo Alto, CA: Tioga Publication Co.; 1983: 137–161.
47. Gentner, D. Structure-mapping: A theoretical framework for analogy. Cognitive Science, Vol. 7: 155–170; 1983.
48. Van Rijsbergen, C.J. Information retrieval. Second edition. London: Butterworth & Co. Ltd; 1979.
49. Hayes-Roth, F. The role of partial and best matches in knowledge systems. In: D.A. Waterman, F. Hayes-Roth, Eds. Pattern directed inference systems. Orlando: Academic Press, Inc.; 1978; 557–574.