Model-Based Indexing and Index Learning in Analogical Design

Sambasiva R. Bhatta and **Ashok K. Goel**
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
bhatta@cc.gatech.edu, goel@cc.gatech.edu

Abstract

Analogical reasoning is the process of retrieving knowledge of a familiar problem (source analog) similar to the current problem (target) and transferring that knowledge to solve the problem. The power of an analogical reasoner thus comes in part from the ability to retrieve the "right" analog when a target is specified. Indexing of analogs therefore is an important issue in analogical reasoning. This issue in fact has three different aspects: (i) indexing vocabulary, (ii) learning of the indices to a new analog, and (iii) use of indices for retrieving stored analogs. We have been exploring the hypothesis that the reasoner's mental models of the analogs give rise to the answers to these issues. We have tested this hypothesis in the context of analogical design of physical devices. In this paper, we describe how structure-behavior-function (SBF) models of devices help in addressing the indexing issues in analogical design. We also describe how the IDEAL system implements and evaluates the model-based scheme to indexing and index learning.

Introduction

Analogical reasoning is the process of retrieving knowledge of a familiar problem (called *source analog*) similar to the current problem (called *target*) and transferring that knowledge to solve the problem. The power of an analogical reasoner thus comes in part from the ability to retrieve the "right" analog when a new problem is specified. Indexing of analogs therefore is an important issue in analogical reasoning.

Actually, the indexing issue has three different aspects: (i) what might be the indexing vocabulary, (ii) how might the indices be learned for a new analog when it is stored in memory, and (iii) how might the learned indices be used for analog retrieval. We have been exploring the hypothesis that mental models of analogs give rise to the indexing vocabulary, enable and constrain the learning of indices for new analogs, and provide similarity measures for matching a target problem with the stored analogs and retrieving relevant ones. We have tested this hypothesis in the context of designing physical devices.

In earlier work, we showed how structure-behavior-function (SBF) models of devices provide the indexing vocabulary and enable the retrieval of analogs relevant to the target problem (Goel, 1992). We also showed how SBF models enable and constrain the transfer of structural knowledge from the source analog to the target problem (Goel, 1991a), and how the concurrent transfer of structural and behavioral (i.e., causal) knowledge leads to the acquisition of the SBF models of new analogs (Goel, 1991b). In this paper, we describe how SBF models enable and constrain the learning of indices to new analogs. We also describe how the IDEAL system implements and evaluates the model-based scheme for indexing and index learning in analogical design.

Analogical Design

IDEAL is an operational system that autonomously designs physical devices such as electrical circuits and heat exchangers. It takes as input a specification of the function of a desired design and the structural constraints on it. The structural constraints may specify, for example, what components cannot be (or must be) used in the design. The system gives as output a specification of a structure that realizes the desired function and satisfices the structural constraints.

A design analog in IDEAL specifies (i) the functions delivered by the stored design, (ii) the structure of the design, and (iii) a pointer to the causal behaviors of the design (the SBF model). Since the input to the system is a specification of the functional and structural constraints on the desired design, the design analogs are indexed both by the functions that the stored design can deliver and by the structural constraints it satisfies, where the delivered functions act as the primary indices.

Given the specification of a design problem, IDEAL retrieves the closest matching analog from the analog memory. The "closeness" of a match is determined by how many features and which features in the problem specification (function and/or structure) are the same as (or different from) the respective indices of a candidate analog. Then IDEAL uses the SBF model of the retrieved design to modify and transfer the design structure to the given problem. It also revises and transfers the causal behaviors of the old design, and thus it generates not only a new design but also a SBF model for it. Then it evaluates the new design by a qualitative simulation of the new SBF model. Finally, IDEAL learns indices for the new design analog as described in this paper and stores the design for potential reuse.

Device Models

IDEAL represents the knowledge of how devices work in the form of structure-behavior-function (SBF) models. SBF models are based on a *component-substance* ontology. In this ontology, the structure of a device is viewed as constituted of *components* and *substances*. Components form the structural topology of the device. Substances *flow* between components with a corresponding *rate*. Substances have *locations* in reference to the components in the device. They also have

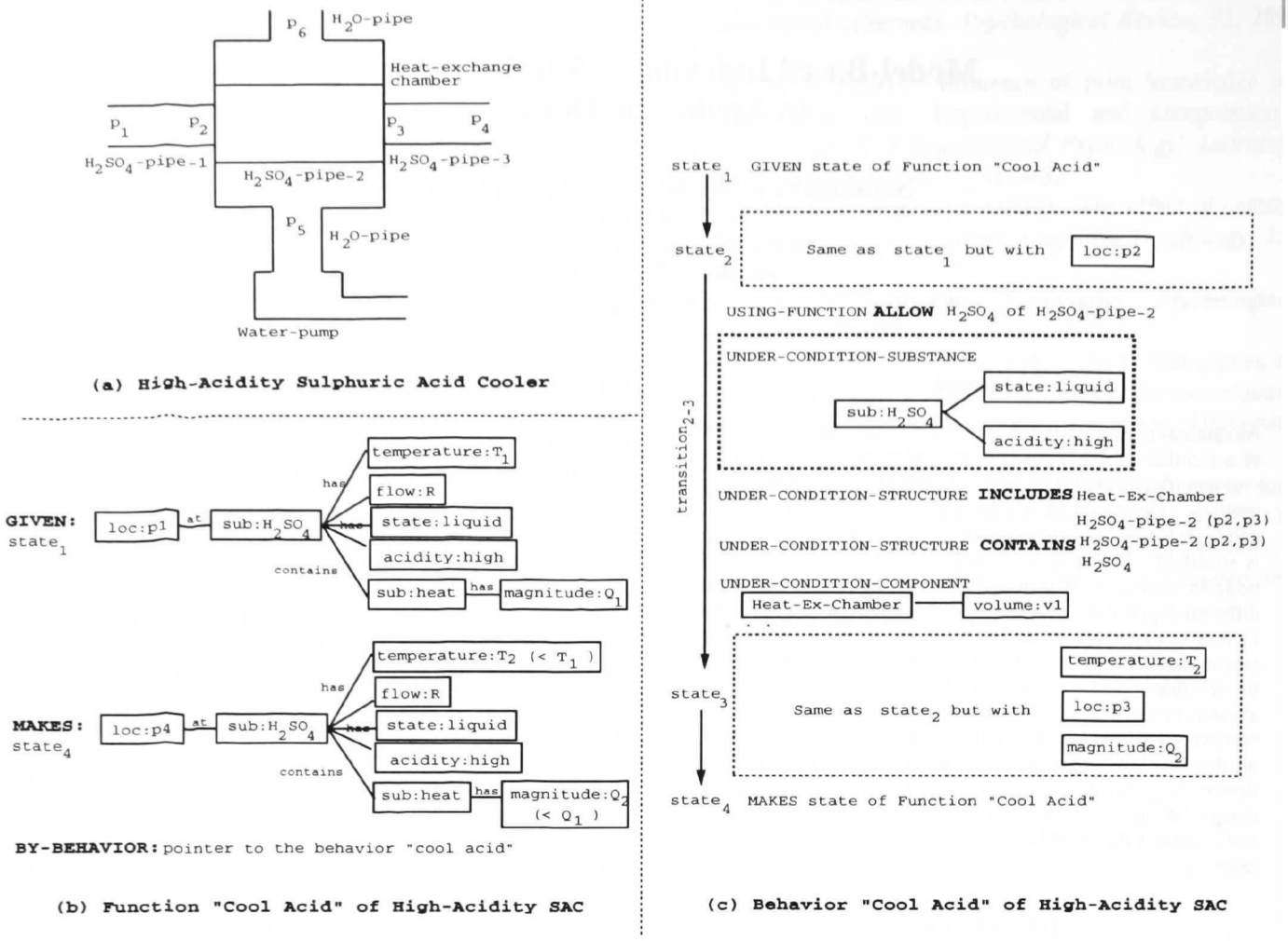


Figure 1: **Design of High-Acidity Sulfuric Acid Cooler**

behavioral properties, such as *acidity* of *sulfuric acid,* and corresponding *values,* such as *low, high,* etc. The constituents of the SBF model are described below.

Structure: The structure of a design is expressed in terms of its constituent components and substances and the interactions between them. Figure 1(a) shows a `high-acidity sulfuric acid cooler` (SAC). Components and substances can interact both *structurally* and *behaviorally.* For example, water can flow from H_2O-pipe to heat-exchange chamber only if they are *connected,* and sulfuric acid flows from p1 to p2 due to behavior *allow* of H_2SO_4-pipe-1. The typology of such structural and behavioral interactions is borrowed from (Bylander, 1991).

Function: A function of a device is a desired output behavior of the device such as cooling of some substance or producing light of certain color and intensity. A function is represented as a schema that specifies the behavioral state the device takes as input, the behavioral state it gives as output, and a pointer to the internal causal behavior of the design that achieves the function. Figure 1(b) shows the function "cool acid" of the `high-acidity SAC`. Both the input state and the output state are represented as *substance schemas.* The input state specifies that sulfuric acid at `location` p1 in the topography of the device (Figure 1(a)) has the properties `temperature`, `flow`, `state`, and `acidity`, and the corresponding values `T1`, `R`, `liquid`, and `high`. It also specifies that the sulfuric acid contains another substance `heat` whose `magnitude` is `Q1`. Similarly, the output state specifies the properties and the corresponding values of the substance at `location` p4. Note that the values `T1` and `T2` of temperature of sulfuric acid are used to denote some corresponding quantitative/qualitative values (e.g., 100 degrees, high, low, etc.) and that `T2` < `T1`. In addition, the slot *by-behavior* acts as an index into the causal behavior that achieves the function of cooling sulfuric acid. The resulting organization of behaviors around the functions they deliver is based on Sembugamoorthy and Chandrasekaran's (1986) functional representation scheme.

Behavior: The internal causal behaviors of a device are viewed as sequences of *state transitions* between *behavioral*

states. The annotations on the state transitions express different kinds of context (e.g., causal, functional, and structural contexts) in which the transformation of state variables, such as substance, location, properties, and values, can occur. The causal context, for instance, provides *causal relations* between the variables in preceding and succeeding states. Figure 1(c) shows a fragment of the causal behavior that explains how sulfuric acid is cooled from `temperature T1` to `T2`. `State2` is the preceding state of `transition23` and `state3` is its succeeding state. `State2` describes the state of sulfuric acid at location p2 and so does `state3` at location p3. The annotation USING-FUNCTION in `transition23` indicates that the transition occurs due to the behavior **allow** of H_2SO_4-pipe-2 (i.e., functional context).

Similarly, the UNDER-CONDITION-SUBSTANCE annotation specifies that the behavior *allow* of H_2SO_4-pipe-2 can allow the flow of only some substances—the substances that are in liquid state and that have high acidity. One of the annotations UNDER-CONDITION-STRUCTURE specifies that the heat-exchange chamber INCLUDES H_2SO_4-pipe-2 (i.e., structural context). Furthermore, the causal behaviors can be specified at different levels of detail. A single transition between two states can be described as a sequence of several states at a different level of detail using the primitive *by-behavior* (not shown in the example).

Model-Based Indexing

Since the SBF models explicitly specify the device functions, the device structure, and the causal principles that underlie the functioning of the device, a design analog can be indexed by any and all of these. But IDEAL's analog memory is organized functionally, i.e., its indexing scheme reflects the reasoning tasks it addresses. Since the design task it addresses is specified by the functional and structural constraints on a desired design, the design analogs are indexed by the functions they deliver and the structural constraints they satisfy. The SBF models provide the vocabulary for this indexing of the design analogs.

Functional Indexing of Design Analogs

For now, we focus on IDEAL's use of device functions for indexing the design analogs—we will return to the use of structural constraints for indexing in the evaluation section. The design analogs are organized in generalization-specialization hierarchies. As described earlier, a function is expressed in terms of substance schemas. Since the substance schema specifies properties of substances, IDEAL uses them as dimensions along which design analogs are generalized/specialized. For example, it organizes designs of acid coolers along the dimension of property *acidity,* and discriminates on the corresponding values `low` *vs* `high` as shown in Figure 2.[1] The HNO_3 `cooler case` in Figure 2(a) is a design of low-acidity nitric acid cooler and hence stored under the category that refers to *low-acidity coolers.*

[1]The figure 2(a) illustrates the analog memory only along the dimension of *acidity* for clarity. The property *acidity* in our example is important because the choice of *pipe* in the design depends on whether it has to allow a low-acidity substance or a high-acidity substance.

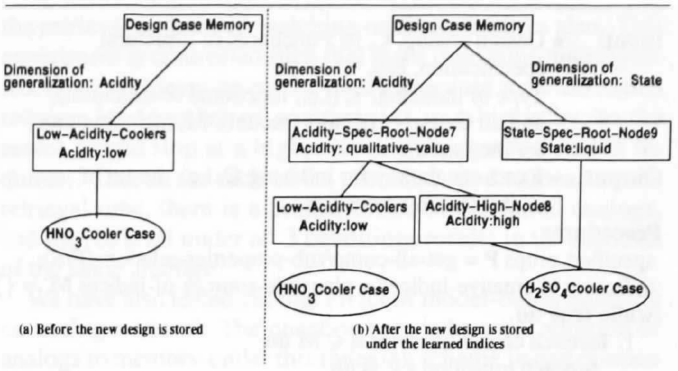


Figure 2: **Snapshots of IDEAL's functionally organized analog memory**

Learning of Functional Indices

Now consider the task of identifying indices for the design of high-acidity SAC (Figure 1) when storing it in memory. Although the analog memory presently has designs of acid coolers organized only along the dimension of property *acidity,* perhaps the new analog should be indexed along other dimensions also so that it is more useful in future design episodes. There are two different issues concerning the selection of indices for the new design analog. First, if a new design is stored only along the substance properties specified in its function, the retriever would not be able to make use of knowledge of other substance properties relevant to the design. Second, if the new design is indexed by all the properties of the substance in its function, then the retriever may retrieve a design based on a match with an unimportant property, which can make analog transfer hard or even impossible. So, the issue becomes how to determine the substance properties that are relevant to the functioning of high-acidity SAC. In general, the issue is how to learn "new" indexing vocabulary.[2]

IDEAL capitalizes on the knowledge of the causal behaviors in the SBF model of the new analog. In particular, it uses the behavioral requirements on the substance expressed under UNDER-CONDITION-SUBSTANCE to identify the substance properties relevant to the functioning of the design. These behavioral requirements of a substance specify that in order for the transition to take place, the properties of the specified substance should satisfy certain conditions.

IDEAL's algorithm for selecting useful indices to a new analog is shown in Figure 3. Given a new design analog and the type of indexing (i.e., functions) this method traverses through the causal behaviors in the SBF model of the new analog to identify substance properties on which the working of the design is predicated. Since the SBF model can specify multiple behaviors, the outer loop (in step 1) in the algorithm analyzes each causal behavior in the model. The second loop is for analyzing the transitions within a causal behavior. If a

[2]By "new" indexing vocabulary, we do not mean that the vocabulary is new to IDEAL but rather it is new for the purpose of indexing.

Input: • Design analog, **C**, its Functional or Structural specification, **F/S**,
Type of indexing, **T**, (i.e., functional or structural), and all Causal behaviors (model), **M**, including one for the function.

Output: • Exact vocabulary for indexing **C**, i.e., the set of useful features from **F/S**.

Procedure:

specified-props **P** = get-all-comp/sub-properties-relations(**F/S**);
indices = alternative-indices = plausible-sources-of-indices **M'** = {};
while true do
1. **foreach** causal behavior **B** ∈ **M do**
   foreach transition **t** ∈ **B do**
   conditions-on-features **CF** = get-under-conditions(**T, t**);
   indices = indices ∪ {f | feature f ∈ **CF** ∧ f ∈ **P**};
   alternative-indices = alternative-indices ∪
   {f | feature f ∈ **CF** ∧ f ∈ **P** ∧ f ∈ parameter-relations(**t**)};
   if indices = **P then** exit(indices);
   if CF = {} **then M'** = **M'** ∪ get-detailed-behavior(**t**);
2. **if M'** = {} **then**
   if indices ≠ {} **then** exit(indices);
   if alternative-indices ≠ {} **then** exit(alternative-indices);
   exit(**P**);
3. **M** = **M'**;
4. **M'** = {};

end.

Figure 3: **The algorithm for obtaining functional indices to design analogs**

substance property is part of the causal context of a transition,[3] then the algorithm adds it to the set of indices if it is a property in the functional specification; and, it adds the property to the set of alternative indices if it is also in the parameter-relations on the transition. Since the causal behaviors in IDEAL's SBF model are specified at different levels of detail, the algorithm searches the space of behaviors in a breadth-first manner. If a higher level behavior does not lead to the identification of any useful substance properties, then the more detailed behavior, indicated by *by-behavior*,[4] is added to the list of plausible sources of indices.

For example, given the function of `high-acidity SAC` and its causal behavior (Figures 1(b) & (c)), the above method results in `acidity` and `state` as the indexing features for storing this analog in memory. This is because the annotation on `transition23` specifies that the transition can occur only under certain conditions on properties `state` and `acidity` of the substance flowing through H_2SO_4-pipe-2. The initial analog memory (Figure 2(a)) did not have the property `state` as part of its indexing vocabulary. The SBF model however suggests that `state` is a useful index to the new design analog, and hence IDEAL indexes the new analog by `state` also. A snapshot of the analog memory after storing this design is shown in Figure 2(b).

Once the indices are selected, IDEAL uses similarity-based learning to generalize them. Under each property, IDEAL organizes the analogs in a binary tree discriminated on values of the property in the analogs. It uses the differences in the values of a given property that constitute a type of functional difference between two designs to determine whether the two designs belong to the same category or to different categories. For example, the design of high-acidity SAC is stored under the category of `Acidity-High-Node8` that is different from that of `Low-Acidity-Coolers` (Figure 2(b)) because their values of acidity differ. The level to which the indices are generalized depends on how similar are the corresponding values in the new and old analogs in memory. For instance, a more general category `Acidity-Spec-Root-Node7` is created that covers both `low` and `high` values of acidity.[5] Note that H_2SO_4 Cooler Case is stored in multiple levels corresponding to the nodes `Acidity-Spec-Root-Node7` & `Acidity-High-Node8` under the property acidity and at one level corresponding to the node `State-Spec-Root-Node9` under the property state.

Evaluation

We have evaluated IDEAL's model-based indexing and index learning along a number of different dimensions—we will discuss only one of these dimensions in detail, and briefly mention the others.

Learning multiple types of indices: In IDEAL, the same representations of SBF models that provide functional indices also provide structural indices. We have tested and found that the same index learning method described in this paper also works for learning structural indices to design analogs. (See (Bhatta & Goel, 1993).)

Learning in multiple domains: In addition to the domains of electric circuits and heat exchangers, we have tested and found that the same method of model-based index learning applies to learning indices to analogs from other domains such as reaction wheel assemblies, controllers, electronic circuits, and electromagnetic devices.

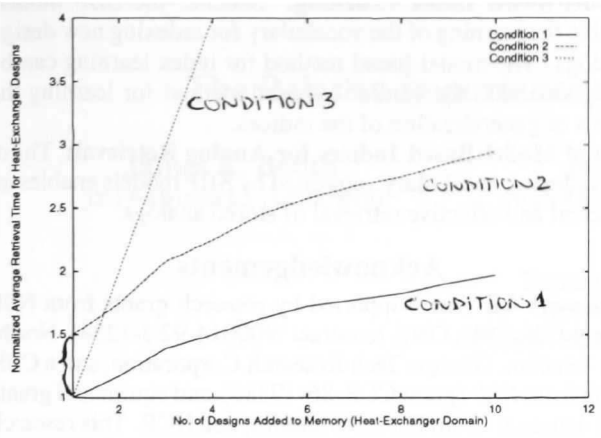
Effect on the Performance Task of Analog Retrieval: We used 20 different designs from two different domains (electric circuits and heat exchangers) to test the effect of model-based index learning on retrieval. The independent variable is the number of analogs added to memory and the dependent variable is the normalized average time for retrieving any of the analogs in memory. The retrieval time is measured in terms of the number of comparisons needed between a specified problem and the stored analogs along each dimension (i.e., property of substance in functional specification) common to the problem and stored analogs. The retrieval time is normalized with respect to the time it takes to retrieve an analog when only that analog is in memory.

As analogs are added to any memory, the subsequent retrieval time increases. In the first set of experiments, the question is whether model-based index learning has any useful effect on the growth of the retrieval time. Within each domain, we compared the retrieval times on 10 problems as

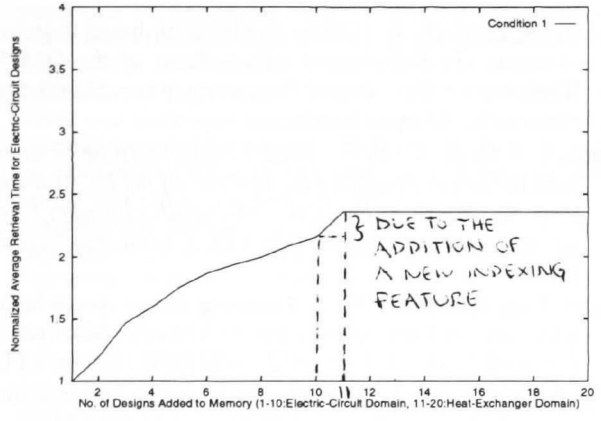
[3] get-under-conditions in the algorithm gets the annotations such as UNDER-CONDITION-SUBSTANCE and UNDER-CONDITION-COMPONENT from the given transition corresponding to the type of indexing used.

[4] obtained by the function get-detailed-behavior in the algorithm.

[5] The general value of acidity at this higher-level node comes from IDEAL's knowledge of qualitative values and quantitative values. If the values are qualitative, it is determined by climbing up a known value hierarchy. And, if the values are quantitative, a new value range is created that spans from the lowest of the two child nodes in the discrimination tree to the highest of the two.



(a) Effect on retrieval under 3 conditions of indexing



(b) Effect of the addition of analogs in one domain on the retrieval in another

Figure 4: **Performance measures on analog retrieval task**

the 10 analogs are added to memory under three different conditions: (1) analogs are stored using model-based index learning, (2) analogs are stored along all possible features in the respective problems (i.e., models are not used to select the relevant indices), and (3) analogs are stored without any organization in memory (i.e., the bottom-line condition in which the retrieval requires an exhaustive search through a list). The results are shown in Figure 4(a). It is evident from the graph in Figure 4(a) that the rate of growth of retrieval time in model-based index learning condition is the slowest. The next best is the condition (2), with the condition (3) being the worst. The reason condition (1) is better than condition (2) is precisely because the SBF models help store the analogs in a relevant, smaller number of features in the problems. In this first experiment, new indices are added to memory when the first analog is stored. The difference in the number of features selected under conditions (1) and (2) is just 1. The difference in the retrieval times in these two conditions when only one analog is stored hence indicates the advantage due to pruning out merely one feature. In condition (2), the retrieval time grows faster as analogs are added because of the addition of analogs along the irrelevant feature(s) which in turn increases the retrieval cost due to matching on those features also. This experiment is controlled such that there is no confounding effect due to retrieval on partial match (because a partial match requires less number of comparisons in a hierarchy, as the search would stop at a higher level, than a perfect match requires). That is, for each of the problems used to measure the retrieval time, there is a perfect match in the stored analogs, and the retrieval under all 3 conditions results in the retrieval of the same analogs.

We have also tested another effect of model-based indexing on analog retrieval. The question here is how the addition of analogs to memory under this indexing scheme in one domain affects the retrieval of analogs in another domain.[6] We first stored the 10 analogs in the domain of electric circuits. There are two features under which IDEAL stored these analogs hierarchically based on the feature values. Then, we measured the normalized average retrieval time on the retrieval of these 10 analogs as 10 more analogs from the domain of heat exchangers are stored. IDEAL stores the second set of 10 analogs under different features than those for the first 10 (as the different sets of features in the problems characterize the domains to be different). The results are shown in Figure 4(b). As evident from the graph in Figure 4(b), the retrieval of analogs in the domain of electric circuits is unaffected with the addition of analogs in the domain of heat exchangers except for the spike in the retrieval time once when a new feature is added to memory as an index. The spike in the retrieval time is due to the comparisons required at the root node in analog memory to discriminate between the features (that is, for instance, to select the hierarchy under *voltage* and weed out the hierarchies under *acidity* and *state*). Thus model-based indexing is effective in grouping analogs based on their content, and this in turn enables retrieval of only semantically relevant analogs for given problems. Although this suggests a useful effect of model-based indexing on the quality of retrieval in restricting the retrieval to a relevant domain, it is yet to be empirically determined how exactly it affects the quality of retrieval within a given domain.

Related Research

The IDEAL system evolves from our earlier work on the Kritik project (Goel, 1991a; 1991b; 1992). IDEAL's SBF models, for example, are directly borrowed from Kritik.

Falkenhainer, Forbus & Gentner (1989) describe the use of mental models for enabling analogical transfer. But they do not address the issue of analog retrieval. Our work on IDEAL suggests that mental models are also useful for addressing the indexing issues in analog retrieval.

Like Kritik and Kritik2, the analog memory in IDEAL is organized functionally, i.e., the indexing depends on the functional requirements of the reasoning task(s). This is a very general organizational principle. Bhatta and Ram (1991), for example, use the same principle to organize scripts and schemas for the task of story understanding.

Since the SBF models provide an explanation of the functioning of a device, our work is also related to explanation-based approaches to learning such as explanation-based generalization (EBG) (Mitchell, Keller & Kedar-Cabelli, 1986),

[6]Two domains are considered distinct if the structural elements (e.g., batteries, pipes) in the domains are different.

explanation-based learning (EBL) (DeJong & Mooney, 1986), and especially explanation-based indexing (EBI) (Barletta & Mark, 1988). This relationship can be analyzed along the dimensions of the learning task, the learning strategy, and the knowledge used by the learning method.

Learning task: The learning tasks in both (Mitchell, Keller & Kedar-Cabelli, 1986) and (DeJong & Mooney, 1986) pertain to concept learning, not index learning. The index learning task addressed by Barletta and Mark (1988) is closely related to and yet different from the one we address. EBI assumes that a pre-enumerated set of indexing features is available, and its learning task is to select some subset of the set of features. In contrast, IDEAL knows only about the types of features that are to be used as indices (e.g., the function) but identifies the exact vocabulary for indices from its SBF model of the new analog. Of course, IDEAL too knows the vocabulary as part of its representations of the SBF model, but does not know a priori the specific vocabulary for indexing. Further, IDEAL learns multiple types of indices (e.g., device function and structural constraints).

Learning strategy: In addition, our model-based scheme differs from EBI in the learning strategy itself, although both integrate explanations and experience for index learning. Firstly, EBI necessarily determines both irrelevant and relevant indexing features. IDEAL in contrast needs to determine only the relevant features. This is because the SBF model generated by its problem solver automatically rules out all irrelevant features. Secondly, IDEAL integrates model-based learning with similarity-based learning (SBL). Specifically, it uses the model-based method to determine the relevant indexing features, and SBL to generalize over the selected features.

Types of knowledge: The explanations in a SBF model are quite different from the explanations in EBG, EBL, and EBI. Firstly, the explanations in EBG, EBL, and EBI are general in that they specify only the form of an explanation as being a resolution proof in FOL (i.e., the explanation can be of any type depending on which context it is used for), whereas IDEAL's explanations are of a specific type (i.e., functional) and the content theory of the explanations is critical to its success. Secondly, the explanations in EBG and EBL specify how an example is an instance of a target concept, and those in EBI refer to the malfunctions of devices, while SBF models are explanations of the normal functioning of devices. Thirdly, the explanations in EBG, EBL, and EBI are constructed at runtime from domain specific rules whereas IDEAL's SBF models are formed by revising old models as part of the problem solving. Fourthly, IDEAL's SBF models are grounded in a well-defined component-substance ontology.

Conclusions

Our experiments with IDEAL lead us to conclude that mental models of analogs provide a useful method for addressing the indexing issues in analogical reasoning. In reference to analogical design in particular, they lead to three specific conclusions.

Model-Based Indexing: First, structure-behavior-function models, together with a specification of the task for which the design analog may be reused, give rise to the vocabulary for indexing design analogs. This vocabulary arises from a deeper domain ontology.

Model-Based Index Learning: Second, the SBF models enable the learning of the vocabulary for indexing new design analogs. The model-based method for index learning can be integrated with the similarity-based method for learning the levels of generalization of the indices.

Use of Model-Based Indices for Analog Retrieval: Third, the indexing vocabulary suggested by SBF models enables an efficient and effective retrieval of stored analogs.

Acknowledgements

This work has been supported by research grants from NSF (grant C36-688), ONR (contract N00014-92-J-1234), Northern Telecom, Georgia Tech Research Corporation, and a CER grant from NSF (grant CCR-86-19886), and equipment grants and donations from IBM, Symbolics, and NCR. This research has benefited from numerous discussions with other members of our research group, in particular Eleni Stroulia.

References

Barletta, R. & Mark, W. (1988). Explanation-based indexing of cases. In Kolodner, J. (Ed.), *Proc. of the DARPA Workshop on Case-Based Reasoning*, pages 50–60, San Mateo, CA. Morgan Kaufmann.

Bhatta, S. & Goel, A. (1993). Model-based learning of structural indices to design cases. In *Proc. of the IJCAI workshop on "Reuse of Designs: An Interdisciplinary Cognitive Approach"*, pages A1–A13, Chambery, Savoie, France.

Bhatta, S. & Ram, A. (1991). Learning indices for schema selection. In *Proc. of the Florida Artificial Intelligence Research Symposium*, pages 226–231, Cocoa Beach, FL.

Bylander, T. (1991). A theory of consolidation for reasoning about devices. *International Journal of Man-Machine Studies*, *35*(4), 467–489.

DeJong, G. & Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning*, *1*(2), 145–176.

Falkenhainer, B., Forbus, K., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, *41*, 1–63.

Goel, A. (1991a). A model-based approach to case adaptation. In *Proc. of the Thirteenth Annual Conference of the Cognitive Science Society*, pages 143–148, Chicago.

Goel, A. (1991b). Model revision: A theory of incremental model learning. In *Proc. of the Eighth International Conference on Machine Learning*, pages 605–609, Chicago.

Goel, A. (1992). Representation of design functions in experience-based design. In Brown, D., Waldron, M., & Yoshikawa, H. (Eds.), *Intelligent Computer Aided Design*, pages 283–308. Amsterdam, Netherlands: North-Holland.

Mitchell, T. M., Keller, R., & Kedar-Cabelli, S. (1986). Explanation-based generalization: A unifying view. *Machine Learning*, *1*(1), 47–80.

Sembugamoorthy, V. & Chandrasekaran, B. (1986). Functional representation of devices and compilation of diagnostic problem-solving systems. In J.Kolodner & C.Riesbeck (Eds.), *Experience, Memory and Reasoning*, pages 47–73. Hillsdale, NJ: Lawrence Erlbaum.