An Analogical Theory of Creativity in Design

Sambasiva R. Bhatta[1] and Ashok K. Goel[2]

[1] NYNEX Science & Technology, 500 Westchester Ave., White Plains, NY 10604.
[2] College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280.

Abstract. Creative design often involves large, complex modifications to the design topology. Making these modifications typically requires transfer of design configurations from different designs to the new problem. We describe an analogical theory of creative design called *model-based analogy* (MBA). In this theory, case-specific structure-behavior-function (SBF) models of past designs enable abstraction of *generic telelogical mechanisms* (GTMs) at storage time. The goal of adapting a specific design to address a new design problem leads to the retrieval of a relevant GTM and its instantiation in the context of the case-specific SBF model at transfer time. Thus GTMs learned from past designs mediate the transfer of abstract design configurations to a new problem.

1 Background, Motivations and Goals

Traditional case-based reasoning provides a process account of adaptive design in which the design modifications are small, simple and local. The process is characterized by reminding of a past design case based on its similarity to a given design problem, and direct transfer of the structure of the design case to the current problem. In earlier work (Goel 1991a), we showed that a Structure-Behavior-Function (SBF) ontology of physical devices provides a vocabulary for indexing known design cases and measures for determining the functional similarity of a past design with a given *function — structure* design problem. We also demonstrated that case-specific, hierarchically-organized SBF models give rise to an array of modification strategies that enable design modifications of two kinds: modifications to the parameters of design elements (i.e., subdesigns that are recursively decomposable into the primitive components and substances) in the structure of the known design, and replacement of design elements by functionally-similar elements. In addition, we showed that the SBF models enable evaluation of modified designs, and, if the evaluation succeeds, their storage in the case memory organized in function-oriented discrimination networks. We call this computational theory *adaptive modeling.*

In this paper, we describe recent work on creative design characterized by large, complex modifications to the topology of a past design. In particular, we describe a computational theory of extracting, abstracting, and composing design configurations from different designs and composing them with a candidate design. In this theory, case-specific SBF models of past designs enable abstraction of generic behavior-function (BF) models. The goal of adapting a specific past design to address a new problem leads to the retrieval of a relevant generic

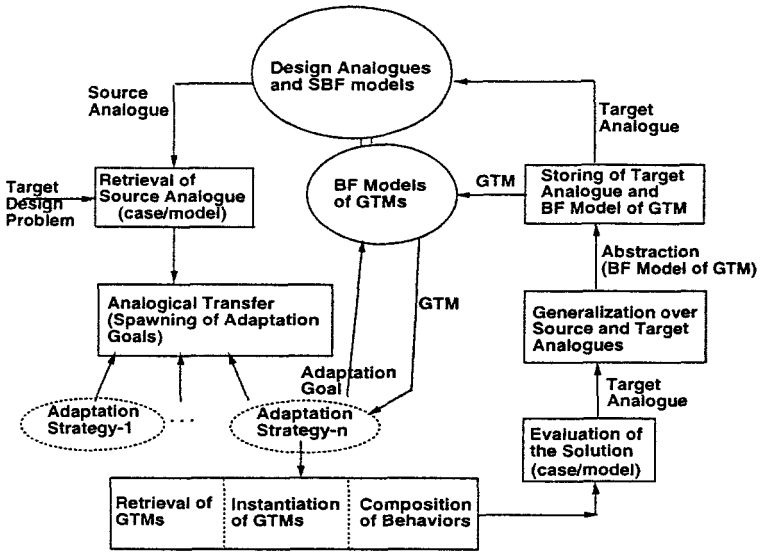

Fig. 1. IDEAL's partial process of Analogical Design via GTMs

model and its instantiation in the context of the case-specific SBF model of the past design. Thus the generic models mediate analogical transfer of design knowledge from familiar designs to the new design problem. We call this theory *model-based analogy* (MBA). IDEAL system instantiates and evaluates the MBA theory.

2 The Process of Model-Based Analogy

The computational process of MBA for design takes as input a specification of the functional and structural constraints on a desired design (i.e., the target design problem), and gives as output a structure (i.e., the solution) that realizes the specified function and also satisfies the structural constraints. MBA also gives an SBF model that explains how the structure realizes that function. A design case in this process specifies (1) the functions delivered by the past design, (2) the structure of the design, and (3) a pointer to the causal behaviors of the design (the case-specific SBF model). It indexes the design cases both by functions that the stored designs deliver and by the structural constraints they satisfy.

Figure 1 illustrates a portion of IDEAL's process of analogical design via one type of generic models called *generic teleological mechanisms* (GTMs). GTMs specify abstract, possibly complex, patterns of relations between output functions and internal behaviors for achieving the functions. They are "mechanisms" because they specify behaviors (or processes), "teleological" because they are in the service of functions, and "generic" because they pertain to classes of device domains, not just to a specific device or a particular device domain. Examples include cascading, feedback, and feedforward mechanisms.

When there are differences between the source and target problems, IDEAL spawns goals for adapting the source design. Different types of functional differences lead to different types of adaptation goals, some requiring only simple modifications (such as local parameter tweaks) and some others requiring more complex modifications (such as large topological changes). In order to control the reasoning involved in making large and complex modifications, IDEAL needs knowledge that can encapsulate the relationships between the modifications and their causal effects. In device design, GTMs provide such knowledge. Therefore, IDEAL uses the knowledge of GTMs in making some types of complex modifications that involve changes to the device topology in the similar past design and thus exhibiting creativity in design. In this paper, we focus on the subtasks of spawning of adaptation goals and achieving them by the use of GTMs.

Case-specific SBF Models. IDEAL represents its comprehension of specific design cases (i.e., case-specific device models) in the SBF language (Goel 1991b). This language provides conceptual primitives for representing and organizing knowledge of the structures, behaviors, and functions of a device. In this representation, the **structure** of a device is viewed as constituted of *components* and *substances*. Substances have *locations* in reference to the components in the device. They also have *behavioral properties*, such as *voltage* of *electricity*, and corresponding *parameters*, such as 1.5 volts, 3 volts, etc. Figure 2(a, b) illustrates the case-specific SBF model of a simple amplifier and Figure 2(c, d) that of an inverting amplifier. For each device, the structure, its function, and the behavior that achieves the function are shown.

A **function** in the SBF models is a behavioral abstraction and is represented as a schema that specifies the behavioral state the function 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. The pair of states indicated by GIVEN and MAKES in Fig. 2(b) shows the function "Amplify Electricity" of the simple amplifier. Both the input state and the output state are represented as *substance schemas*. Informally, the function specifies that the amplifier takes as input electricity of voltage V_{in} volts (i.e., 1) at i/p and gives as output electricity of voltage V_{out} volts (i.e., 100 ± 20 where 100 is the average value and 20 is the fluctuation around the average value) at o/p.

The internal causal **behaviors** in the SBF model of a device explicitly specify and explain how the functions of structural elements in the device get composed into device functions. The annotations on the state transitions express the *causal*, *structural*, and *functional contexts* in which the transformation of state variables, such as substance, location, properties, and parameters, can occur. Figure 2(b) shows the causal behavior that explains how electricity applied at the input location i/p of the simple amplifier is transformed at the output location o/p. The annotation USING-FUNCTION in $state_2$ — $state_3$ indicates that the transition occurs due to the primitive function "ALLOW electricity" of op-amp.

Generic Models. IDEAL represents its GTMs using the same SBF language as above. The SBF representation of a GTM encapsulates two types of knowledge:

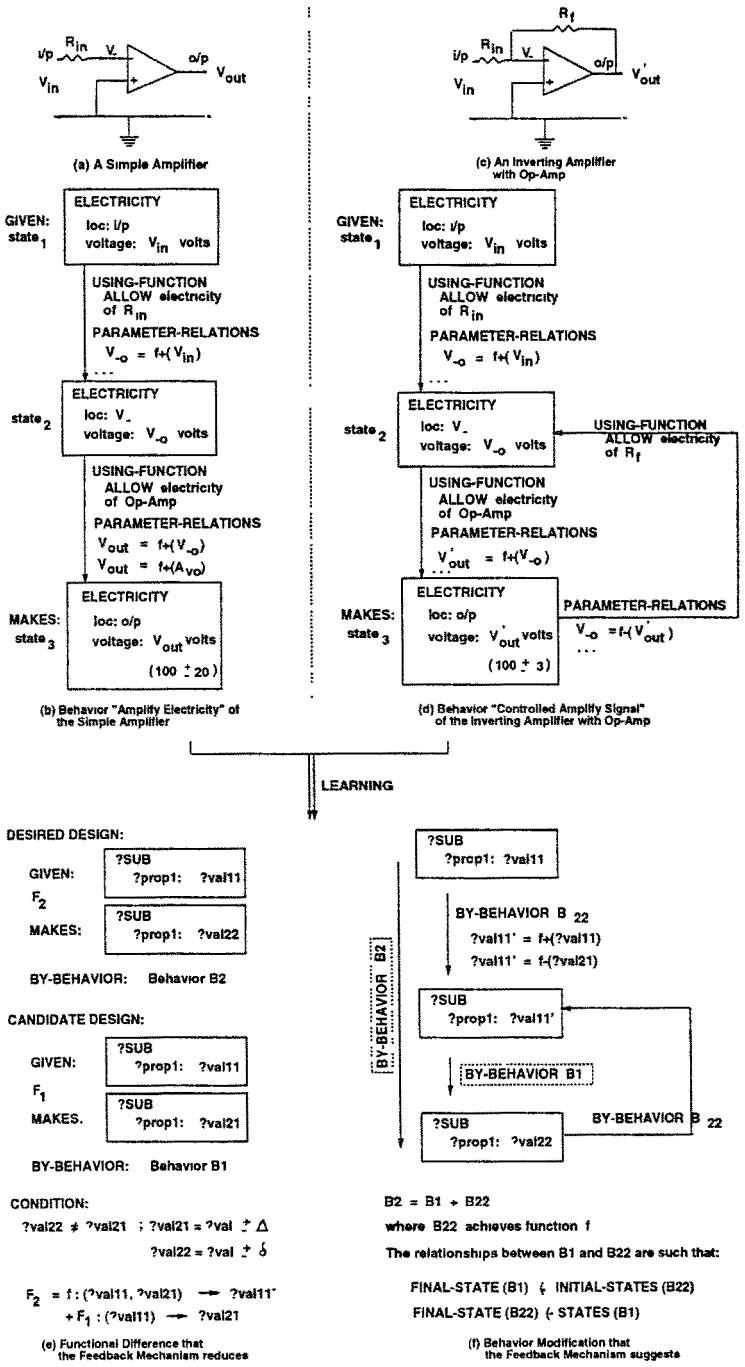

Fig. 2. Learning of Feedback Mechanism

knowledge about the patterns of differences between the functions of known designs and desired designs that the GTM can help reduce; and knowledge about patterns of modifications to the internal causal behaviors of the known designs that are necessary to reduce the differences. That is, it specifies relationships between patterns of functional differences and patterns of behavioral modifications to reduce those functional differences. For example, Fig. 2 (e) & Fig. 2(f) respectively show these two types of knowledge for a partial model of the feedback mechanism.[3] Figure 2(e) shows the patterns of functions F_1 and F_2 respectively of a candidate design available and the desired design, and the conditions underwhich the mechanism is applicable. Because of the tasks for which they are used in MBA, the GTMs are indexed by the patterned functional differences such as shown in Fig. 2(e) (i.e., the fluctuations in the output substance property values are large vs. small). The model of the feedback indicates that the desired behavior (B_2) can be achieved by modifying the candidate behavior (B_1) through setting up the indicated causal relationships between the latter and the additional behaviors (that achieve the subfunctions of F_2 other than F_1 characterized in the conditions of the mechanism). In particular, the feedback mechanism suggests to add a causal relationship from a change in the output substance state to a change in an earlier state (input state or intermediate state) in the candidate behavior so that the effective input to the device is modified. Figure 2(f) shows (both diagrammatically and textually) the relationships in the generic model of the feedback mechanism that IDEAL learns from the two designs of amplifiers.

3 Acquisition of GTMs

Suppose that IDEAL's case memory contains the design of a simple amplifier (Fig. 2(a, b)). The output of this device is dependent on the open loop gain (A_{Vo}, a device parameter) of the op-amp and is typically very high (ideally ∞) and unstable. Consider that IDEAL is given the problem of designing an amplifier whose function (Fig. 2(d)) is to deliver a specific, controllable output electricity (which does not fluctuate much), i.e., an output electricity of voltage V'_{out} volts ($= 100 \pm 3$ where 100 is the average value and 3 is the fluctuation allowed around it) given an input electricity of V_{in} ($= 1$) volts. IDEAL uses the specified function as a probe into its memory of design cases and retrieves the design of the simple amplifier because the two functions are similar. Suppose now that IDEAL only has a simple strategy such as replacing a component in a past design to deliver new functions. Given the model of the simple amplifier shown in Fig. 2(b), IDEAL cannot localize the modification to reduce the difference between the source and the target and hence it fails. Then, if an oracle presents the correct design (whose structure is schematically shown in Fig. 2(c) and the case-specific SBF model in Fig. 2(d)), IDEAL can learn a generic model of the feedback mechanism (Fig. 2(e, f)). For details, see (Bhatta and Goel, 1996).

[3] Feedback can be open loop or closed loop. The feedback mechanism described here is one type of closed-loop feedback in which the output substance, fedback substance (i.e., controlling substance), and the input substance are all same.

4 Analogical Transfer via GTMs

Consider now a design problem IDEAL solves in the domain of mechanical controllers. Suppose that the new problem specifies a function that given the substance angular momentum of magnitude L_i and clockwise direction at an input location (gyroscope), the device needs to produce the angular momentum of magnitude L_o' proportional to the input and the same direction at a specified output location. It also specifies the constraint that the output cannot fluctuate much around an average value (i.e., $L_o' = L_{avg} \pm \delta$, where δ is small). This is the problem of designing a gyroscope follow-up (Hammond 1958).

Suppose also the design of a device (Fig. 3 (a, b, c)) which transfers angular momentum from a gyroscope to an output shaft location is available in IDEAL's case memory (or is given explicitly as part of the *adaptive* design problem). This device's function is that given an input angular momentum of magnitude L_i and clockwise direction at the input (gyroscope) location, it produces a proportional angular momentum of magnitude L_o and of clockwise direction at the output shaft location; however, L_o fluctuates over a large range, i.e., $L_o = L_{avg} \pm \Delta$, where Δ is large. IDEAL retrieves (if not given explicitly) this design because the desired function matches with this design's function.

Now, IDEAL's task is to modify the available design to deliver the desired function. Simple modifications such as replacing a component in the design case will not result in a device that solves the new problem because there is no single component in the device that seems responsible for the large fluctuations and that which may be selected for modification. Then the issue is if and how IDEAL can solve such a non-local adaptation problem.

The first step for IDEAL in abstraction-based analogical transfer is to retrieve the GTM. It uses the difference in the functions of the candidate and desired designs as a probe into its memory because it indexes the mechanisms by the functional differences and the decomposability conditions on the desired functions. It retrieves the feedback mechanism because the current functional difference, namely, the fluctuation in the output property is large vs. small (i.e., Δ *vs.* δ), matches with the difference that the feedback mechanism reduces which is specified in a device-independent manner. Then, it tries to match the decomposability condition on the desired function in the feedback mechanism (see Fig. 2(e) for the condition $F_2 = ...$) with the desired function in order to find the subfunctions f (or g) that need to be designed for and composed with the candidate function. By performing this match, as guided by the SBF language, IDEAL finds the subfunction f:(L_i, L_o') — L_{ww}', i.e., it needs to design for a structure that takes two inputs. angular momentum of magnitude L_i and angular momentum of magnitude L_o', and gives as output an angular momentum of L_{ww}' in the opposite direction at the location of pivot in the candidate design.

The next step for IDEAL in this process is to transfer the retrieved GTM to the target design problem by instantiating it in the context of the problem. When the abstractions are GTMs, this process involves designing for the subfunction(s) determined by matching the applicability conditions of the mechanism and com-

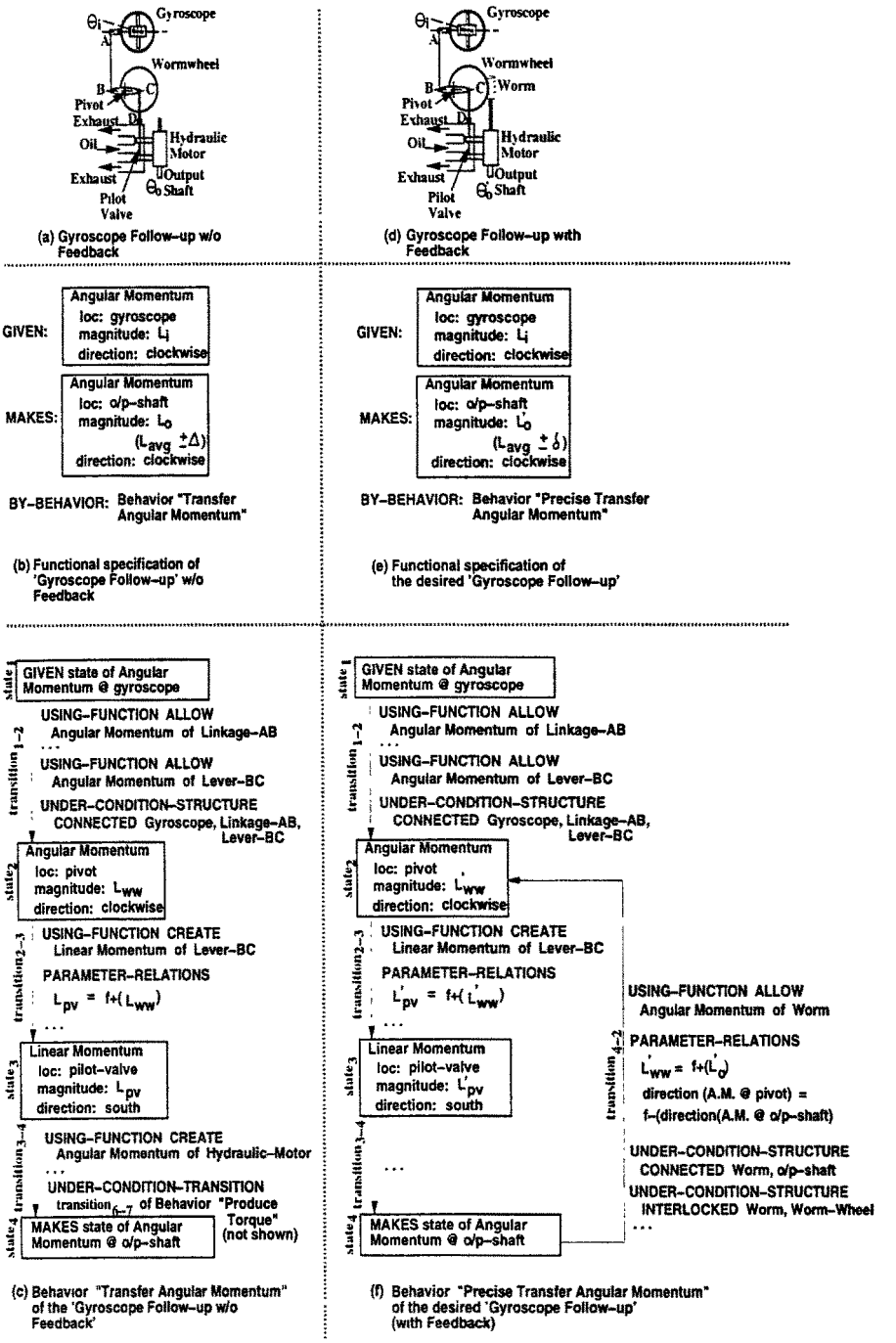

Fig. 3. The two designs of Gyroscope Follow-up *before* (a, b, c) and *after* (d, e, f) instantiating the Feedback Mechanism

posing the new sub-behavior(s) with the behavior of the candidate design as per the relationships specified in the retrieved mechanism. In the current scenario, the subfunction IDEAL needs to design really has two parts (as it takes two inputs and produces one output): one for transferring angular momentum from the input location to the pivot location, and the other for transferring angular momentum from the output shaft location to the pivot location. The first part is already designed for in the candidate design as the behavior segment $state_1 \rightarrow state_2$ (Fig. 3(c)) achieves it. Therefore, in successfully instantiating the mechanism in the candidate design of gyroscope follow-up, IDEAL only needs to find a behavior (and a structure) that accomplishes the second part.

Consider the concrete scenario from IDEAL in which it has the knowledge of a component (called *worm*) whose function is to transfer an input angular momentum to an output location with the magnitude proportional to the output component and the direction dependent on the direction of threading on the worm. This component reverses the direction of the input angular momentum. IDEAL retrieves that component because the desired part of the subfunction matches with its function. It substitutes the appropriate parameters in the behavior of the retrieved design (i.e., worm) to generate a behavior for the desired subfunction. Then it composes that behavior (i.e., B_{22}) with the behavior of the candidate design (i.e., B_1) as per the specification of the causal relationships in the feedback mechanism (as in Fig. 2(f)) to propose a behavior (shown in Fig. 3(f)) for achieving the desired function. Note that the resulting modification in the design of gyroscope follow-up is a non-local modification because the topology of the candidate design changed. Thus the instantiation of GTMs can enable non-local modifications in device design and in turn creativity in design.

5 Evaluation

We conducted several experiments with IDEAL in a number of dimensions described below in order to evaluate its theory of model-based analogical design.
(1) *Generality in terms of domains:* We tested IDEAL in four different domains, namely, the domains of simple electric circuits, heat exchangers, electronic circuits, and mechanical devices (including momentum controllers and velocity controllers) for both learning and use of GTMs. IDEAL could both learn and use the GTMs in the different domains.
(2) *Computational feasibility and efficacy:* We tested IDEAL with twelve distinct pairs of designs (i.e., source and target) from these four different domains for learning and use of six different GTMs (i.e., cascading, four types of feedback, and one type of feedforward). In all these cases, IDEAL was successful in learning GTMs, and in accessing and using them in solving design problems. The largest design in IDEAL had about 10 structural elements.
(3) *Generality in terms of representations:* IDEAL uses the same SBF language to represent both the case-specific models of devices and the case-independent models of GTMs.

(4) *Generality in terms of tasks:* IDEAL addresses multiple tasks, for example, both the learning and the use of GTMs in model-based analogical design.

6 Related Research

Much of the past work on case-based design has been limited to adaptive design in which the design modifications are small, simple and local. The relatively little research that explored creative design has focused on organization and exploration of case memory. IM-RECIDE (Gomes et al. 1996) and IMPROVISER (Wills and Kolodner 1996) are two recent examples of goal-directed exploration of the case memory. IDEAL's case memory is organized in multiple function-oriented discrimination networks. Also, each case is multiply indexed, both by the functional requirements and the structural constraints satisfied by the stored design. Case retrieval is goal-based while case storage is model-based; in the storage phase, the SBF model of the new design enables index learning.

Case-based theories typically involve direct transfer of the structure of familiar designs to new design situations. But in some case-based design systems, high-level abstractions do play an important role, especially in case indexing and case reminding. For example, KRITIK used functional abstractions of the stored design as case indices (Goel 1991a), and DEJA VU used them for hierarchical organization of the case memory (Smyth and Cunningham 1992). As mentioned earlier, IDEAL indexes cases by the functional requirements and structural constraints of the stored designs, and organizes them in discrimination networks based on a taxonomy of functions.

DSSUA (Qian and Gero 1992) is a recent analogical design system based on the notion of design prototypes. Like GTMs, design prototypes too specify functional relations and causal structures in a class of devices, but, unlike a GTM, a design prototype also specifies the generic physical structure of the device class. While a design prototype is a generalization over design cases such that a case is an instance of a prototype, a GTM is an abstraction over design prototypes such that a design prototype is a subclass of a design pattern. DSSUA uses an analogical process similar to that of the structure-mapping engine (SME) (Falkenhainer et al. 1989) to abstract causal behaviors at transfer time. In contrast, IDEAL abstracts GTMs at storage time for potential reuse. GTMs are indexed by the problem-solving goals stated in terms of functional differences between two design situations. This aspect of MBA shares the perspective of purpose-directed analogy (Kedar-Cabelli 1985).

Some recent work in case-based reasoning has explored the learning of adaptation knowledge from cases. For example, Leake (1995), and Hanney and Keane (1996) describe alternative methods for acquiring adaptation rules from adaptation cases. IDEAL too learns adaptation knowledge from cases (Bhatta and Goel 1996). Unlike other work on adaptation learning, the adaptation knowledge that IDEAL learns is declarative and demonstratably domain-independent.

7 Conclusions

Creative design involves large and complex modifications to the topology of known designs. These modifications are enabled by transfer and composition of design configurations from different designs to the new problem. MBA is a computational theory of this kind of analogy-based creative design. In MBA, case-specific SBF models of past designs enable abstraction of GTMs at storage time. The goal of adapting a specific similar past design to address a new design problem leads to the retrieval of a relevant GTM and its instantiation in the context of the case-specific SBF model of the past design. GTMs thus mediate the transfer of design knowledge across domains.

Acknowledgments

This paper has benefited from numerous discussions with members of the Intelligence and Design research group at Georgia Tech. This work has been supported in part by research grants from NSF (IRI-92-10925 and DMI-94-20405) and ONR (research contract N00014-92-J-1234).

References

Bhatta, S., Goel, A.: From design experiences to generic mechanisms: Model-based learning in analogical design. AI EDAM **10** (1996) 131–136

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

Goel, A.: A model-based approach to case adaptation. In Proc. of the Thirteenth Annual Conf. of the Cog. Sci. Soc. (1991a) 143–148

Goel, A.: Model revision: A theory of incremental model learning. In Proc. of the Eighth Intl. Conf. on Machine Learning (1991b) 605–609

Gomes, P., Bento, C., Gago, P., Costa, E.: Towards a case-based model for creative design. In Proc. of 12th European Conf. on AI (1996)

Hammond, P. H.: Feedback Theory and Its Applications. (1958) The English Univ. Press Ltd., London, UK.

Hanney, K., Keane, M.: Learning adaptation rules from a case base. In Proc. of Third European Workshop on Case-Based Reasoning (1996)

Kedar-Cabelli, S. T.: Purpose-directed analogy. In Proc. of 7th Annual Conf. of the Cog. Sci. Soc. (1985) 150–159

Leake, D.: Combining rules and cases to learn case adaptation. In Proc. of 17th Annual Conf. of the Cog. Sci. Soc. (1995)

Qian, L., Gero, J. S.: A design support system using analogy. In J.S. Gero, editor, Proc. of the Second International Conference on AI in Design (1992) 795–813

Smyth, B., Cunningham, P.: Deja vu: A hierarchical case-based reasoning system for software design. In Proc. of 10th European Conf. on AI (1992)

Wills, L., Kolodner, J.: Towards more creative case-based design systems. In *Case-Based Reasoning: Experiences, Lessons, and Future Directions*, D. Leake (Ed.), (1996) 81–91. AAAI/MIT Press