Submitted to *Artificial Intelligence*. Draft of September 18, 1990. Comments welcome.

Analogy in Context

Brian Falkenhainer
System Sciences Laboratory
Xerox Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto, CA 94304

Abstract

Context appears in two forms in analogy. First, there is the context in which the analogy is performed. Second, each statement being compared is done so within the context of each analogue's overall description. By explicitly taking context into account, we can provide a more robust account of analogy. Central to this account is that each element of an analogue's description has an identifiable *role*, corresponding to the dependencies it satisfies or its relevant properties in the given context. The relations comprising each analogue are placed in correspondence by virtue of their filling corresponding roles. This work makes three principle contributions. First, it extracts a set of core principles underlying reasoning from similarity which enables correspondences between syntactically distinct expressions and the controlled introduction of many-to-many mappings. Second, it uses these principles to provide a unifying look at a variety of recent analogical and case-based reasoning systems. Third, it introduces the *map and analyze cycle*, a general computational framework for computing similarity correspondences and producing useful inferences. The work is demonstrated with implemented examples, including classification, causal explanation, and a number of examples from the literature.

1 Introduction

When plotting the perfect murder, how is an icicle analogous to a knife? When packing an ice-chest, how is an icicle analogous to an ice-cube? Icicles, ice cubes, knives, and spoons have numerous properties. Their relative similarities vary according to the context in which the similarity is assessed. Analogical reasoning is a process in which similarities between two situations are identified and used to suggest that knowledge of one situation may also apply to the other. Yet, by what criterion should similarity be judged? How does similarity lend credibility to this kind of knowledge transfer?

Analogical similarity is typically defined in terms of matching patterns in the descriptions of two analogues, seeking both isomorphic structures and features described by the same terms [16, 21, 25, 33, 44]. Thus, if our icicle and knife were explicitly described in their respective narratives as hard and pointed objects, their similarity would

be identified via these common terms. This view of analogy is amenable to simple computational frameworks and has demonstrated some success on a variety of tasks. It rests on the fundamental assumption that corresponding aspects of two analogues are represented in the same manner and may be identified using simple, syntactic matching criteria. As task complexity and representational sophistication grow, this assumption breaks down. As we will demonstrate, similarity criteria based on one-to-one mappings between identical terms are too restrictive and brittle. Furthermore, existing solutions based on the conceptual closeness of terms [5, 27, 43, 44] (e.g., generalization hierarchies) are too weak; they can produce unmotivated and incorrect similarity correspondences. Determining similarity requires understanding which aspects of the terms being compared are relevant in a given context. This insight was crucial to the success of PHINEAS [12, 13, 14], a program that constructs causal explanations of observed phenomena based on their similarity to understood phenomena. As we will show, similar insights may be found in several other recently developed systems as well [3, 30, 32, 34].

This paper uses the insights gained from recent efforts to reexamine the principles underlying analogical similarity. It describes *contextual structure-mapping* (CSM), a knowledge-level account of similarity that attempts to explain the basic principles by which reasoning from similarity is meaningful and identify the implicit assumptions embodied in existing systems. CSM provides a general characterization of similarity that allows the controlled introduction of many-to-many mappings and matches between syntactically distinct expressions. This account is based on the notion that each element of an analogue's description has an identifiable *role*, corresponding to the dependencies it satisfies or its relevant properties in the given context. The relations comprising each analogue correspond by virtue of their filling corresponding roles. The notion of role includes a component's participation in a given behavior, design decisions achieving a design's rationale, an actor's actions and traits achieving a story's plot, and an antecedent justifying its consequents in a logical proof. For example, in the context of a murder mystery, an icicle is analogous to a knife in the role of being an effective murder weapon because they have the requisite properties, not because they are conceptually close along some dimension. In the context of packing an ice chest, different properties become relevant and an icicle may become analogous to an ice cube.

The second part of this paper shows how contextual structure-mapping serves to unify and explain some of the intuitions behind a variety of recently developed approaches, including derivational replay [6, 37], tweaking [30, 29], knowledge-based pattern matching [3], and my work on analogical explanation [12, 13]. It shows that each technique is an attempt to exploit some aspect of role and context in analogy.

The third part of this paper uses this characterization of similarity to examine the computational requirements of an analogical reasoning system. It describes the *map and analyze cycle*, a general computational framework for computing correspondences and producing useful inferences. Initial correspondences between the two analogue's given descriptions are found using simple pattern matching operations. These correspondences are extended by analyzing incomplete portions of the analogy, seeking ad-

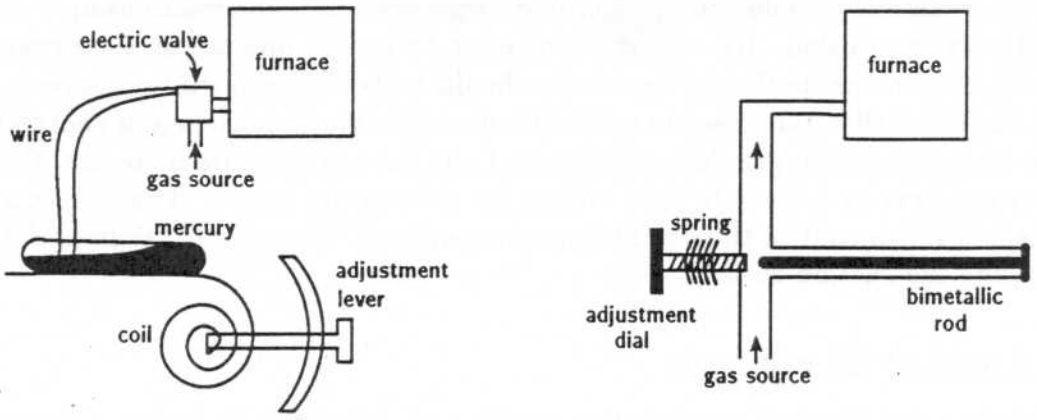

Figure 1: Two thermostats.

ditional information about the roles of unmatched items. An implementation using the *structure-mapping engine* (SME) [15, 16] is described.

1.1 Objectives

While noting similarities between two situations is a central aspect of analogy, we are most interested in its use for problem solving, in which knowledge of one analogue (the *base*) is used to answer questions about the other (the *target*). Analogy begins when a base analogue that has potentially relevant similarities to the target is retrieved (or directly supplied, as by a teacher). Ultimately, it may lead to memory reorganization, comparative evaluation of competing hypotheses, and continued use of the base analogue to further explore its consequences for the target domain. This paper is concerned with what lies in between – how similarities are elaborated and used to solve an open problem once a candidate base analogue has been identified.[1] It presents a general model, applicable to across domain comparisons (with within-domain comparisons as a special case), that may be used to either accelerate problem solving or produce plausible inferences to overcome an incomplete domain theory.

For example, consider the thermostat shown in Figure 1(a). This device regulates a heater to maintain a specified temperature. Temperature deviation is sensed by the coil, which expands and contracts as it heats up and cools down, respectively. The furnace is on when the mercury in the glass tube electrically connects the two terminals, which keeps the valve open. As the environment temperature increases, the coil expands and the glass tube's angle increases. Eventually, the temperature reaches a point where the mercury moves to the right, disconnecting the terminals and closing the valve. The

[1]See [13, 14] for a description of PHINEAS, a complete analogical reasoning system that uses CSM in conjunction with retrieval and hypothesis evaluation mechanisms.

lever position controls the device's goal temperature by shifting the relationship between temperature and position. By analogy, how does device (b) operate as a thermostat? Intuitively, our understanding of device (a) should make this explanation easier. Like the coil, the bimetallic rod senses temperature deviation via expansion and contraction. When in the on position, gas flows past its end. As the temperature increases, the rod lengthens and acts as a valve to shut off the gas flow to the heater. The dial acts like device (a)'s lever to control the equilibrium temperature. The spring stabilizes the dial by increasing the friction on its threads.

1.2 Analogy in context

What are the principles and methods that enable the explanation of device (a) to assist, and perhaps make possible, the process of explaining device (b)? Clearly, one aspect is identifying possible correspondences between their descriptions. In comparing two analogues, most methods only match expressions that use the same predicate (e.g., the two `furnaces`). However, differences between cases or domains often lead to different predicates describing analogous concepts. If changes are to be allowed in response to differences in the two analogues' descriptions, what constitutes an acceptable change? The common approach is to measure conceptual closeness, using a generalization hierarchy [44, 5, 25, 43] or a-priori similarity score [27]. When comparing the two thermostats, this approach might suggest a correspondence between device (a)'s lever and device (b)'s dial (both adjustment knobs). However, it might also suggest a correspondence between device (a)'s coil and device (b)'s spring (both springs). Yet, their similarity is irrelevant and misses the coil's role with respect to the device's components and teleology. The coil's relevant aspect in this context is its thermal expansion characteristics – the same aspect that is relevant to understanding device (b)'s bimetallic rod. When an expression is used in some context (e.g., in a chain of inference), it denotes certain characteristics about the world important for that context. Without knowing which effects are relevant, there is no way to know which generalizations or similarities are appropriate. Thus, *objects and relations are identified as being similar by examining their roles in their respective analogue descriptions.* To do this, we must clearly understand what a role is, how role information affects similarity assessment, and how incomplete role information affects the validity of an analogically derived conclusion.

In addition to predicate similarity, constraints on analogical mapping tend to require that the mapping be one-to-one. However, in device (a), the coil and the valve provide temperature sensing and gas control, respectively, while both functions are provided by device (b)'s rod. Here, an isomorphic mapping fails to fully capture the correspondence; a many-to-one mapping from {`coil, valve`} to {`rod`} is needed. Due to function sharing, many-to-many mappings are common in physical systems. Yet, allowing these mappings can dramatically increase computational cost and lead to incoherent mappings. Thus, the conditions under which the one-to-one restriction should be relaxed must be precisely defined. *These conditions come from explicitly considering the roles of compared items.*

If an expression has more than one role, then each role may lead to a different analogical correspondent.

Finally, several mappings may be possible and some criteria must be used to select the "best" one. The thermostats are relatively unambiguous, but at least two interpretations are conceivable: one in which the coil is mapped to the spring and one in which the coil is mapped to the rod. Evaluation criteria proportional to match size, such as Gentner's [21] *systematicity*, could prefer the former interpretation if the thermostats' descriptions included detailed descriptions of the coil and spring. However, the purpose of the analogy in this context is to explain the devices' overall behavior as thermostats, and only the second interpretation achieves this. *Current goals should have a strong influence over which matches are preferred.*

The process of computing similarities is traditionally depicted as a form of pattern matching between base and target descriptions. However, in realistic memories, not all that is inferrably known about the base and target is explicit in their initial descriptions. Furthermore, they may not be described using the same terminology (e.g., `coil` and `rod` rather than `sensor`). Thus, the process of elaborating correspondences and adapting elements of the base to fit the target situation often requires inferring additional information in response to mapping impasses. Because pattern matching alone can be expensive, and there are many (potentially infinite) inferences that can be made to extend either analogue, this process must be tightly focused and goal directed. Therefore, we decompose it to form a *map and analyze* cycle (Figure 2): use simple matching criteria to determine the best, initial mapping between the analogues, analyze the results and seek additional relevant information about unmatched areas, reexamine the mapping to determine the information's impact on the mapping (i.e., extensions or complete shifts), analyze the new results, etc. In this manner, only when impasses arise, such as an expression having no apparent correspondent in the other analogue, is the domain theory consulted and more detail about that expression and its role sought. Addition-

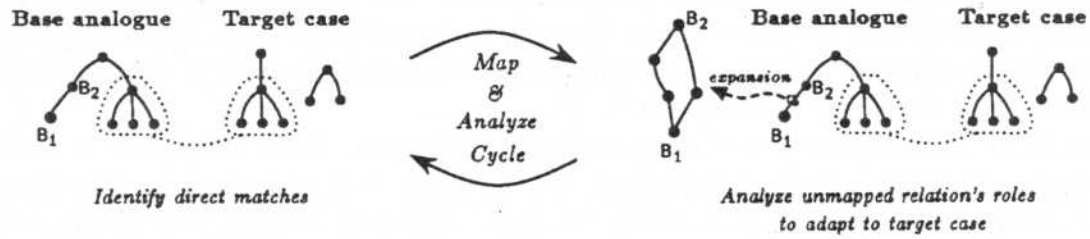

Figure 2: Map and analyze cycle. Analogy involves comparing two representation structures. Obvious correspondences may motivate additional correspondences according to the corresponding elements' roles. The task is complicated by abstracted or implicit reasoning steps in the analogue descriptions, which often must be further analyzed to decompose an expression's roles into finer levels of detail. The map and analyze cycle focuses more in-depth reasoning only where and when needed to converge to a point where the match can no longer be extended.

ally, the separation means that these inferencing processes are only used in furtherance of a single, initially best mapping, rather than the potentially much larger set of possible mappings.

1.3 Assumptions

Each analogue is a representational structure consisting of a set of interdependent *expressions*. An expression may be a predicate-calculus formula, a feature in a feature vector representation, or a node or link in a semantic network. Take $\mathcal{B}$ and $\mathcal{T}$ to denote the base and target structures, respectively. These may represent either actual cases or generalized prototypes, but are always ground. Theory Th ($\mathcal{B}$,$\mathcal{T} \subset Th$) consists of all available knowledge and may supply additional information about the two analogues beyond what is stated in the first two inputs. Analogical inference is then of the form

P_b, Q_b	$(P_b, Q_b \in \mathcal{B})$
$P_b \rightsquigarrow Q_b$	P_b is predictive of Q_b
P_t	$(P_t \in \mathcal{T})$
$P_b \sim P_t$	P_b is similar to P_t
$\overline{\qquad}$	
Q_t	

P and Q need not be correlated in the underlying domain theory in an antecedent/consequent manner (as in $P_b \rightarrow Q_b$). Analogical inference may give the effect of forward inference, abductive inference, and in general pattern completion.

This paper's central focus is on understanding the similarity relationship $\sim$ and how that affects the conclusion Q_t. Some correspondences may be directly identifiable from the base and target descriptions; others arise as a side effect of adapting base information to apply to the target case. Although our account is task-independent, our examples and the computational method described in Section 4 focus on explanation and classification tasks. This class of problems takes three inputs: (1) a description of the target situation and the set of query facts Q_t to be explained (e.g., observations or category membership); (2) a potentially analogous base example and the corresponding set of facts Q_b it explains;[2] and (3) theory Th.

Two fundamental questions arise in analogical inference. First, what does it mean to say that two things are similar? Second, how do these similarities increase the likelihood of inferred facts? Several factors contribute to increasing confidence in analogically derived hypotheses, including logical justification [10], consistency [28], and empirical utility [12, 23]. The fundamental, but not sufficient, criterion for plausibility requires that inferred facts be somehow correlated to those participating in the similarity relationship. This correlation may be causal (P causes Q), functional ($Q = f(P)$), empirical,

[2]The base represents the dependency structure (i.e., proof) containing Q_b. Sometimes the base is simply a set of ground atomic facts $B_1, \ldots, B_n$ describing something satisfying Q_b (e.g., a concept exemplar). In this case, we may view it as a dependency structure of length one, with $Q_b \leftarrow B_1, \ldots, B_n$.

etc, but must be predictive. For example, similarities between U.S. and British politics may suggest that they have similar economies, due to the strong correlation between political and economic policy, but does not suggest they drive on the same side of the road. Inferential strength depends on the specific type of correlation (e.g., material implication is stronger than a weak empirical association). Additionally, it depends on other mitigating factors, such as analogue differences directly providing negative evidence.

Although there have been recent attempts to formally capture the correlation underlying analogical inference (e.g., [1, 2, 9, 10, 35]), much work is still needed. Rather than attempt to fully address this issue here, we separate the criteria that *suggest* plausible inferences from the criteria that *evaluate* them. At this time, the existence of a correlation is taken as sufficient evidence for suggesting an inference. We assume a representation where dependencies between facts are readily discernible (e.g., semantic nets, schemas, frames, dependency and derivation structures). Accepting or rejecting conjectures is relegated to the performance element, which can compare a conjecture's strength to the task's requirements.

2 The role of role

If the roles of an analogue's elements are fundamental to how they participate in an analogy, then what is a role and how is it identified? "Role" appears in various forms across many domains and tasks. In physical systems, a component's role is how it contributes to the system's overall behavior. In design, the role of particular design decisions and artifact components is the satisfaction of particular design specifications and rationale. The role of an agent's actions (as in planning or a story) may be in support of certain outcomes. In deductive proof, the role of an antecedent is to provide logical support for its consequent. For these intuitions to be useful, we need a more precise definition of role that transcends these various domains and tasks. This is found in the logical notion of dependency – an expression's roles correspond to the dependencies it satisfies.

Within an analogue, if Q is a predication whose truth is dependent on predication P, then the *role* of P in that analogue is to satisfy the dependency relationship with Q. P may fill other roles in the analogue as well, in as much as P satisfies other dependencies. P's complete set of roles with respect to analogue $\mathcal{A}_i$ is represented as

$$\text{ROLE}_{\mathcal{A}_i}(P) = \{Q_1, \ldots, Q_n\}$$

and consists of all of the elements of $\mathcal{A}_i$ that P influences.[3] For the remainder of this paper, we will leave the important conditional "with respect to $\mathcal{A}_i$" unstated and drop the subscript.

[3]The contents of $\mathcal{A}_i$ are a function of the purpose of the analogy, e.g., relevant to answering the given query. Thus, $\mathcal{A}_i$ and $\mathcal{A}_j$ may describe the same case but for different purposes.

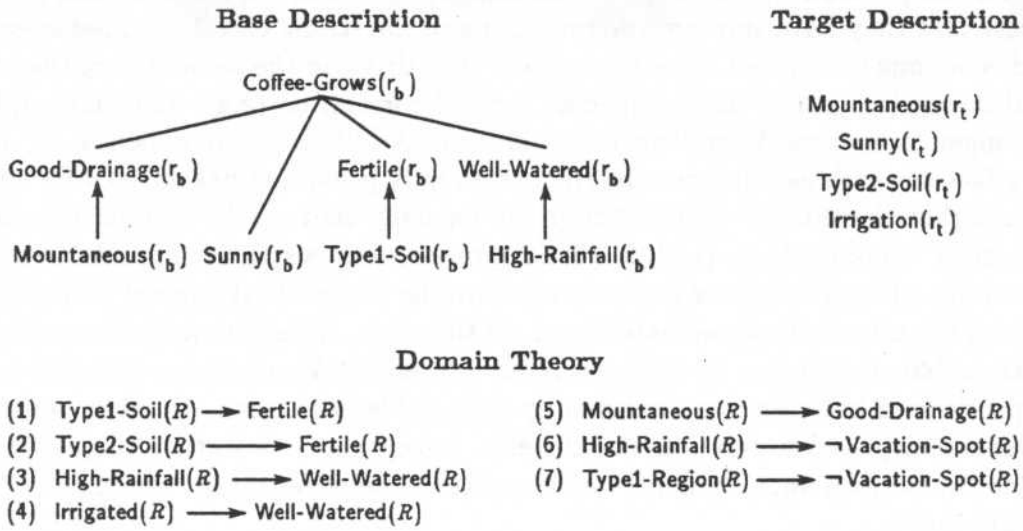

Figure 3: Can coffee grow in region r_t?

The elements of two analogues' descriptions correspond by virtue of having corresponding roles (rather than because they share a number of possibly irrelevant properties). If the dependencies supported by base expression P_b may be satisfied by target expression P_t, then P_b and P_t are said to be *functionally analogous* and may be placed in analogical correspondence. For example, the property `Rainfall` should map to the property `Irrigation` if the role of these conditions is to ensure that a given crop receives sufficient water. Note that they are not functionally analogous in other roles, such as washing a plant's leaves. This may be generalized (and form a recursive definition) by stating that given two corresponding roles (i.e., not necessarily identical), their role fillers may be considered functionally analogous and eligible for being placed in correspondence, independent of predicate (or feature) identicality. The more that is known about the roles of an analogue's elements, the greater the ability to go beyond syntactic similarity measures and produce creative analogies.

For example, suppose we are asked if coffee can grow in region r_t and are given a base case describing region r_b where coffee is known to grow (adapted from [2]). Figure 3 shows the initial base and target descriptions, and the relevant parts of the domain theory.[4] The base case indicates that coffee's ability to grow is a function of climate, sun exposure, and terrain. To draw the analogical inference, the relevant similarities between r_b and r_t must be established. Clearly, the two regions are both *Mountaneous*

[4]In this particular example, *Coffee-Grows*(r_t) follows from the domain theory. The next section discusses how analogy may provide speedup in such circumstances. The subsequent section discusses analogy in the context of a weak domain theory (e.g., *Coffee-Grows*(r_t) is not entailed by the domain theory).

and *Sunny*. Syntactically, the two regions have no other similarities. However, by noting rainfall's role in the base case

ROLE(*High-Rainfall*(r_b)) = {*Well-Watered*(r_b)}

rule (4) indicates that *Irrigated*(r_t) is functionally analogous to *High-Rainfall*(r_b). Likewise, *Type2-Soil* corresponds to *Type1-Soil* via the common role of *Fertile-Soil*. Therefore, the conclusion *Coffee-Grows*(r_t) may be drawn.

2.1 Compiled and abstracted representations

When solving complex problems, if a similar problem has been solved before it is intuitively better to draw on that experience than to start from scratch. While not true for our simplistic example about growing coffee, this is one potential benefit of analogical reasoning. Unfortunately, representations that are adequate for traditional approaches to problem solving may not be suitable for performing analogical reasoning. AI systems tend to represent a minimalist approach to encoding knowledge, in which detailed descriptions or intermediate reasoning steps are avoided to promote efficiency. For example, a physical process may be modeled at a fixed level of abstraction and a design plan may not contain the rationale behind the decisions it embodies. Indeed, knowledge compilation is a central goal of explanation-based generalization [11, 36]. While potentially effective for accelerating reasoning, a great deal of information is intentionally absent from the resulting representations. This poses a significant problem for adaptation of a base analogue to a novel case: these intermediate or second-order justifications are often needed to ascertain the requisite role information (Figure 4).

For example, consider the following schema used to explain why coffee grows in region r_b:

Coffee-Region(r_b)
PRECONDITIONS *High-Rainfall*(r_b) ∧ *Fertile-Soil*(r_b) ∧ *Sunny*(r_b) ∧ *Mountaneous*(r_b)
EFFECTS *Coffee-Grows*(r_b)

This schema is not directly applicable to a situation in which *High-Rainfall* is false and *Irrigated* is true. Why is the *High-Rainfall* precondition there? Suppose a deeper explanation is retrievable and reveals that this condition is important in this context because it ensures that the region is well watered. Without knowledge of the role of *High-Rainfall*(r_b), it would be impossible to justify considering *High-Rainfall*(r_b) functionally analogous to *Irrigated*(r_t).

To address the compiled knowledge problem, the schema's internal justifications must be accessible (either cached or reconstructable). This information may then be consulted as needed during analogical mapping to decompose and elaborate the reasons underlying a particular dependency relationship.[5] In the implementation described in Section 4, a

[5]Including all background knowledge in the initial base and target descriptions would be too expensive. Additionally, which added details are needed cannot be identified until impasses arise during the similarity assessment.

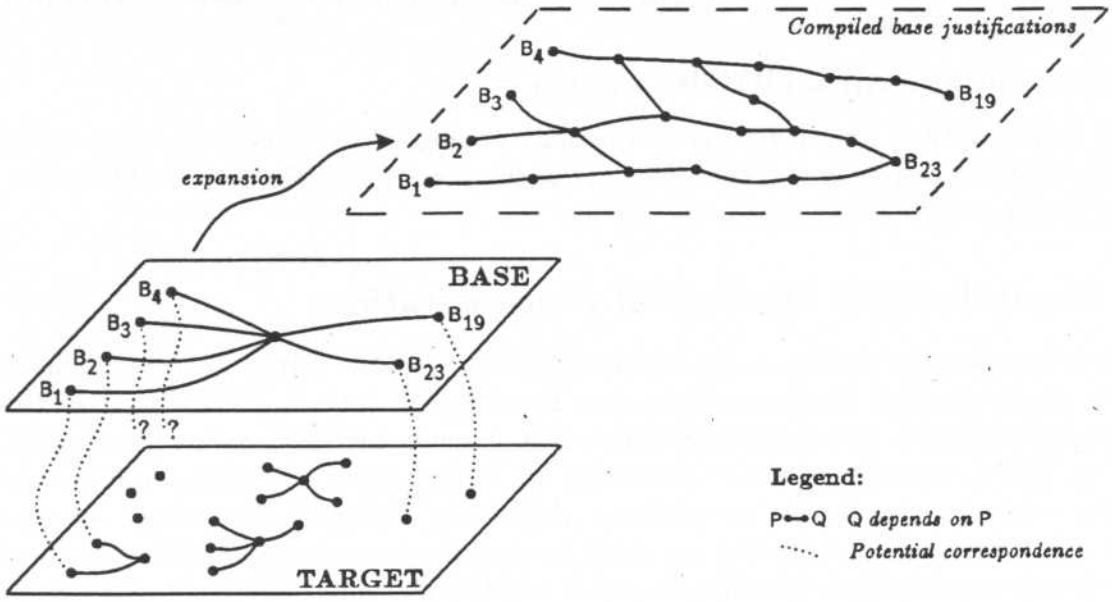

Figure 4: Compiled knowledge problem. Intermediate reasoning steps removed to increase problem solving efficiency are often required when considering how to elaborate analogical correspondences and adapt a solution to unanticipated cases. The **BASE** representation depicts a macro formed by removal of the intermediate justifications within the dashed box. A question mark indicates a relevant base expression having no known target correspondent without more information about its role in the base context.

CACHE field accompanies all compiled schemas. For example, the coffee growing schema should have an added *CACHE* field to store compiled reasoning steps like

$$\textit{High-Rainfall}(\tau_b) \rightarrow \textit{Well-Watered}(\tau_b)$$

This form of role analysis presents a means for sound adaptation of a syntactically inapplicable compiled theory. It offers an extension to explanation-based learning and schema application methods: demand-driven, sound adaptation of an overly-specific macro. Partial retention of a macro's details, and the corresponding space / efficiency tradeoffs, is a topic of future research.

2.2 Where is the inductive leap?

Role analysis as described above uses sound methods over a complete domain theory to establish the logical equivalence of two features with respect to the analogues' dependencies. This form of the functionally analogous relation assumes complete knowledge of all relevant features and complete knowledge of a feature's relevance. However, analogy clearly has ties to induction and can be used to draw plausible conjectures in the presence of a weak, incomplete domain theory.

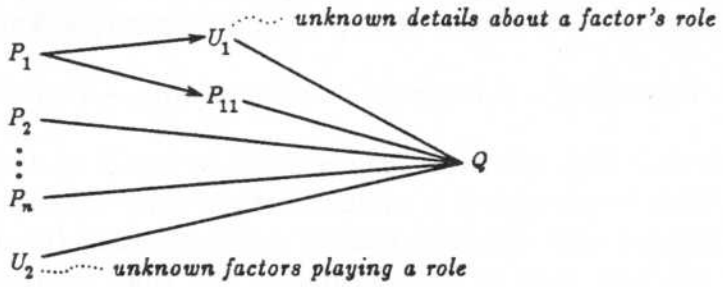

Figure 5: Uncertainty arises due to incomplete role information. This has two forms: (1) partial knowledge of a feature's roles and (2) incomplete knowledge of all relevant features.

One way plausible inferences arise is by making added assumptions about the target to complete an explanation. This is abductive inference, and occurs in many tasks independent of whether analogy is used. For example, when applying the coffee growing schema to region r_t, if the status of *Mountaneous(r_t)* is unknown, it might be assumed in order to make the inference proceed.

More specific to analogy, uncertainty arises due to incomplete role information (i.e., uncertainty about the contents of ROLE(P) and the manner in which they are affected by P). This may be illustrated by considering a simple, abstract base description, in which Q was observed and $P_1 \wedge \cdots \wedge P_n$ was offered as an explanation. Uncertainty then appears in two general forms, as illustrated in Figure 5:

1. Inability to fully state the roles of each relation in the analogues' descriptions (thus, inability to fully state corresponding relations' equivalence with respect to the current context).

2. Inability to state that all of the relations relevant to the desired conclusion are captured by the analogue description (thus, inability to state all the factors that play a role in influencing the conclusion).

In the first case, the antecedent information in the base analogue may be fully predictive (i.e., $P_1 \wedge \cdots \wedge P_n \rightarrow Q$), but more detailed knowledge of their roles is incomplete. Knowledge of an expression's roles provides the detail that enables valid adaptation (i.e., generalization) to novel cases. Incomplete knowledge of an expression's roles introduces uncertainty into this step and reduces it to plausible inference. To make the presence and magnitude of this uncertainty explicit, it must be captured in the underlying domain theory's representation of dependencies. We use "$\rightsquigarrow$" to describe partial role information.[6] Thus, in the derivation of Q, we would have $P_1 \rightsquigarrow P_{11}$, or

$$\text{ROLE}(P_1) = \{P_{11}, U_1\}$$

[6]We are currently examining probabilistic and default logic interpretations for this relation.

where U_1 represents the set of unknown roles for P_1.

This may be demonstrated by reconsidering the coffee growing example:

$$\textit{High-Rainfall}(r_b) \wedge \textit{Tropical}(r_b) \wedge \textit{Sunny}(r_b) \wedge \textit{Mountaneous}(r_b) \rightarrow \textit{Coffee-Grows}(r_b)$$

Previously, we stated that high rainfall is only relevant by virtue of its ensuring a well watered region (i.e., both "water supply is a relevant effect" and "it has no other relevant effects" are true). Suppose now that high rainfall may have other (possibly unknown) attributes that are relevant, such as high humidity, and thus *Coffee-Grows* may not be universally true for the more general case of well watered regions. There may be something intrinsically important about high rainfall beyond supplying ground water (such as humidity) that leads to successfully growing coffee. In this case, the probability that the role of *High-Rainfall*(r_b) is $\{$ *Well-Watered*(r_b)$\}$ is less than one and thus region r_t's analogy to r_b may fail due to the uncertainty in the generalization that enables *Irrigated* to map to *High-Rainfall.*

In the second case, the base description is merely plausible (probabilistic). Either the grain size of the known factors P_i is insufficient to deterministically make predictions, or some relevant factors U_2 are missing from $P_1, \ldots, P_n$. For example, suppose that although coffee is known to grow in region r_b, its rainfall, region type, sun exposure, and terrain are only partially predictive of other regions. Region r_b may have an additional, critical attribute that is either unknown or has unknown relevance, such as its altitude. The validity of conclusions about r_t via similarities to r_b is limited due to incomplete knowledge of which similarities are relevant.

These two sources of uncertainty provoke a number of questions that affect the soundness of an analogical inference: Is base condition B_i needed in the target case? What fortuitous side effects (whose relevance may not be recognized) does base condition B_i cause? What negative side effects (whose relevance may not be recognized) does target condition T_i cause?

2.3 Where is the generalization?

Analogy is often characterized as a generalization process, in which a common abstraction is found between the two analogues. Such generalizations, when they occur, appear in two forms: (1) generalization of individual features (e.g., *prince* to *royalty*) or (2) abstraction of the analogue as a whole (e.g., *Macbeth* to *Shakespearean tragedy*). Here we only consider the first case.[7]

[7]Automated abstraction (as opposed to using existing abstractions) is a little understood, difficult task in which an analogue's objects and relations are all generalized in concert. It is mentioned here only for completeness: the generalization must respect the interdependencies (i.e., the role structure) of the relations being generalized. For example, finding a common abstraction between "the jungle" and "wall street" must maintain the basic interactions between the agents while generalizing the interaction and agent types. The elements cannot be generalized independently.

Identifying similarities between two instance descriptions often requires finding a common generalization for syntactically distinct features. For example, successful completion of the coffee growing example required generalizing *High-Rainfall* and *Irrigated* to *Well-Watered*. However, each feature will typically have many consequences (most being irrelevant) and thus many possible generalizations. For example, high rainfall may generalize to a climatic condition, a source of water supply, a falling object, or an air cleanser. Which line of generalization is allowable depends on the feature's role in the context of the analogue's overall description. Without some source of role information, there is no justification for generalization (e.g., arbitrary generalization failed frequently when attempted in PHINEAS [13]). There are three general ways to determine a feature's role.

First, role information may be explicit in an analogue's description. As shown in the previous two sections, this can be used to provide either valid or plausible generalization. This is the only approach implemented in Section 4.

Second, empirical typicality information may suggest a feature's role prior to its appearance in a particular analogue. For example, given "95% of the time, high rainfall is mentioned in the context of supplying water", there is a 0.95 *a priori* probability that for any analogue $\mathcal{A}_i$ in which *High-Rainfall* appears, the following is true

$$\text{ROLE}_{\mathcal{A}_i}(\textit{High-Rainfall}(r)) = \{\textit{Well-Watered}(r)\}$$

Third, knowledge of the application domain may be used implicitly by restricting the range of possible generalizations to being a priori relevant. We say that a generalization hierarchy has the *task-relevant property* if for every pair of nodes in the hierarchy, they share a common parent if and only if they have a common role in all situations that will ever be encountered by the application (i.e., the hierarchy always reflects an item's purpose). If a hierarchy satisfying the task-relevant property can be constructed, then similarity assessment via proximity in the hierarchy is a computationally attractive alternative to full role analysis. First, it reduces the storage requirements of explicitly maintaining all role information. Second, each pairwise similarity comparison can be computed in time linear in the depth of the hierarchy. Because the task-relevant property is a fallible idealization, it is important to be explicit about when it is being assumed.

A number of systems successfully operate by matching terms that share a common parent in an *ISA* hierarchy [5, 25, 33, 43, 44]. Each appears to satisfy the task-relevant property; in each case, the hierarchy is both *shallow* and *relevant* to the system's single task domain. For example, CHEF [25] builds recipes in the domain of Szechwan cooking. Its hierarchy is a shallow forest (rather than a single tree) in that broccoli and snow peas are both vegetables, but vegetables and meat are not both listed as organic objects. Its hierarchy is relevant to the system's goals in that ingredients are organized along the lines of taste, texture, and style.

3 A unifying look

Understanding the role of role information in similarity assessment provides significant explanatory and unifying power. This section describes several recently developed systems from the contextual structure-mapping perspective and examines the basic assumptions under which they operate. These systems display various aspects of the same basic need to know the role of the individual expressions in each analogue's descriptive context. All of the examples shown have been implemented using the CSM algorithm described in Section 4 and are presented from that perspective. It is important to note that for all of the systems discussed, only a subset of their operations are related to CSM. For example, each system addresses the additional problem of retrieving useful base cases.

3.1 Mapping immutable structures

CSM was inspired by extensive computational experience with Gentner's Structure-Mapping Theory (SMT) [20, 21, 22] and attempts to understand and rectify its limitations. We share the intuition that analogy is about systems of interrelated relations rather than collections of independent facts. Furthermore, several aspects of the CSM implementation (Section 4) are based on the original SMT implementation [15, 16]. However, experience has shown that for some tasks, SMT's restriction that mappings must be one-to-one and between expressions using the same predicate is too strong, and its reliance on representational form is too brittle [13].

While algorithmically very different, the approaches of Gentner [21, 16] and Greiner [23] both effectively reinstantiate a base structure onto a new target case, allowing token (object) and function substitution. Because they assume the system of relations from the base must apply *as is* to the target case, reformulation and role analysis does not come into play. The basic assumption under which analogical inferences are made is that there exists a predictive correlation between the elements of the base structure such that commonality with respect to some suggests commonality with respect to the others. This correlation is captured by the interconnections in SMT's structural representation. In Greiner's framework, it is captured by *abstractions* (precollected sets of relations) and an "`A` is like `B`" hint from a trusted source.

3.2 Derivational replay

Derivational replay mechanisms make the observation that it is typically easier to reuse the problem solving process of a prior episode than the episode's final solution [6, 37]. Taking a stored problem solving plan, a new problem is solved by replaying the plan top-down, resolving subgoals that no longer apply in the current situation. This is intended to provide problem solving speed-up (by borrowing a rule-invocation ordering from a past experience) rather than plausible conjecture or learning. No explicit ana-

logue comparison is performed; rather, rules are assumed to reinstantiate to the target's analogous items. Violation of this assumption can lead to backtracking [37]. Additionally, it means that correspondences are not recorded to ensure consistent reuse across different portions of the problem solving process, which may lead to a lack of focus.

CSM's definition of *functionally analogous* captures the underlying intuition behind replay's appeal over mapping only a final solution: the root problem in reusing a prior solution is the need to understand the role of each part of the solution. Specifically, good adaptability comes from being able to recognize alternate ways to achieve equivalent functionality. In problem solving, adapting a prior solution to a new problem instance is greatly simplified if decisions' rationale are known, so that their intent can be satisfied without necessarily adhering to the same decisions. Top-down replay achieves this by giving the effect of reseeking each role filler in the new situation. An alternate method would start at the solution, and work backwards, analyzing role information at solution transfer impasses. Thus, the implicit assumption in replay mechanisms is that it is more efficient to work forward, replaying the entire decision-making process, than to work backward, reconsidering the decisions (alternate ways to achieve functionality) where needed. This tradeoff is influenced in part by the connectivity of the solution: good modularization is amenable to local modifications with minimal global repercussions. Poor modularization may affect the entire solution. Derivational replay mechanisms probably work best for situations where modularity is poor.

3.3 Explanation reuse

PHINEAS [12, 13, 14], Kedar-Cabelli's PER [32], and the Yale SWALE project [30, 29] are efforts to use analogy to reduce the cost and increase the creativity of explanation building. These systems all use operations that draw upon role information to adapt to differences in the two analogues. In contrast to derivational replay, they compare the situation descriptions first and adapt only those parts of the base explanation that fail to map. This approach rests on implicit assumptions of analogue proximity and good modularity – if the analogues are highly similar, then their instance descriptions should have high overlap and require little adaptation. Identification of this overlap can initiate the analogy process and provide focus by isolating the relations requiring further role analysis. It is also a side-effect of their explanatory task, which cares only about identifying the target's facts that allow inference to the conclusion. In the planning tasks typically addressed by derivational replay, the analogues' differ both in their conclusions (goals) and in their antecedents (starting states). This makes intermediate steps between the starting and goal states more likely to need adaptation, and makes the base's intermediate solution procedure as relevant as its start or goal state descriptions.

CSM was originally developed in support of PHINEAS [12, 13, 14], a program which constructs qualitative explanations of time-varying phenomena based on their similarity to understood phenomena. When there are structural differences between the recalled and observed situations, it examines elements' functional roles to identify similarities

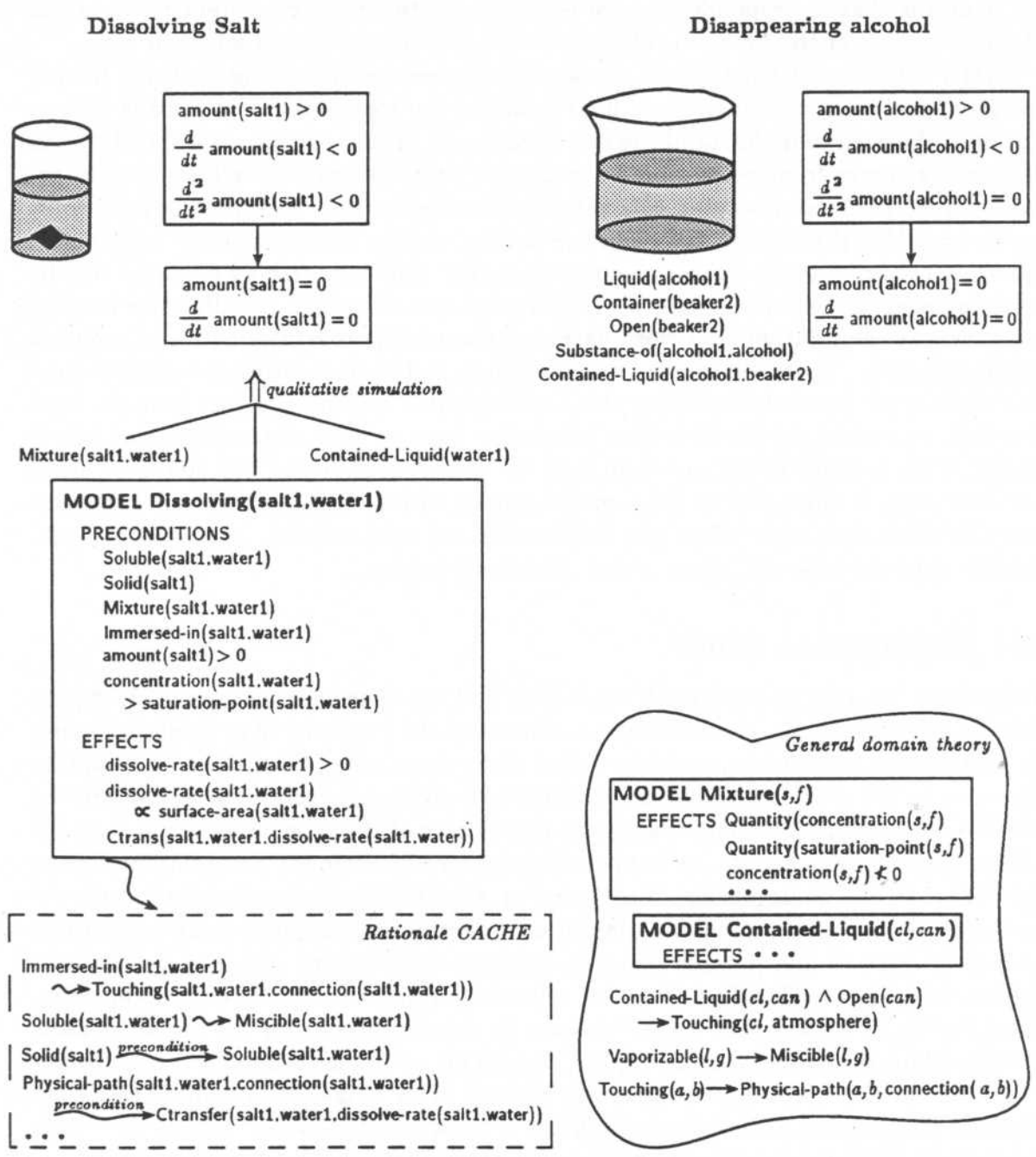

Figure 6: Explaining why alcohol is disappearing from an open container by analogy to salt dissolving in water.

and debug its models. For example, PHINEAS uses an analogy to salt dissolving in a glass of water to explain an observation of alcohol evaporating from an open beaker (Figure 6).[8] Initially, it compares the two situations and finds a possible correspondence between the disappearing salt and the disappearing alcohol. However, it cannot apply the dissolving model to the disappearing alcohol case. First, the dissolving model describes the interactions between salt1 and water1, yet there is no apparent correspondent for water1 in the alcohol scenario. Second, some of the conditions for dissolving are inapplicable to the alcohol scenario. For example, substituting `alcohol1` for `salt1` suggests the precondition `Immersed-in(alcohol1,?unknown)`. Yet, the first argument to `Immersed-in` must be a solid while `alcohol1` is a liquid. To adapt to these differences, PHINEAS examines unmapped elements' roles. The relevant aspect of `Immersed-in` is that it ensures physical contact between the salt and the water (e.g., as opposed to preventing exposure to the air). Seeking similar functionality in the alcohol case, it finds that `Touching(alcohol1,atmosphere)` follows from the alcohol being in an open container. This suggests that `atmosphere`, previously absent from the alcohol scenario's description, may be the missing correspondent for `water1`. With this, PHINEAS reexamines the mapping and finds that the alcohol/atmosphere combination has the requisite properties for the other unfilled roles (e.g., `Vaporizable(alcohol1,atmosphere)` is functionally analogous to `Solube(salt1,water1)`). The mapping phase concludes with a new model applicable to the disappearing alcohol case. PHINEAS then uses qualitative simulation to compare the model's predictions to the observation and revise the model if necessary. Not addressed in this paper are these subsequent evaluation and revision processes, which find that the proposed explanation leads to an interesting set of predictions about unobserved rates and concentrations (see [13]).

PHINEAS contains a sophisticated pattern-matching mechanism (SME [16]) capable efficiently comparing large descriptions of multiple objects. As the example illustrates, PHINEAS' mapping component is able to revise the base and target descriptions with additional facts and objects in response to mapping impasses. However, its use of role information is domain-specific and is limited to behavioral and causal information about structural components.

Kedar-Cabelli [32] describes a method for *purposive explanation replay*, PER. Given a plan schema and an old object to which the schema has applied, PER's task is to find an object in the current state that may be substituted and still achieve the plan. PER replans around preconditions not supported by the current situation. These operations seek functionally analogous expressions in the two analogues by attempting to find alternate inferential support for unmatched items. In one example, the task is to classify a given styrofoam cup as an instance of *Hot-Cup* (Figure 7). The base case is a description of a previously classified ceramic cup. PER begins by seeking each of the

[8] The base representation contains the expression $q_1 \propto q_2$. This is *qualitative process theory* (QP) [18] syntax indicating that q_1 is *qualitatively proportional to* q_2. All else being equal, q_1 increases when q_2 increases and decreases when q_2 decreases. Additionally, *Ctrans(s,d,r)* indicates continuous transfer from *s* to *d* at rate *r*.

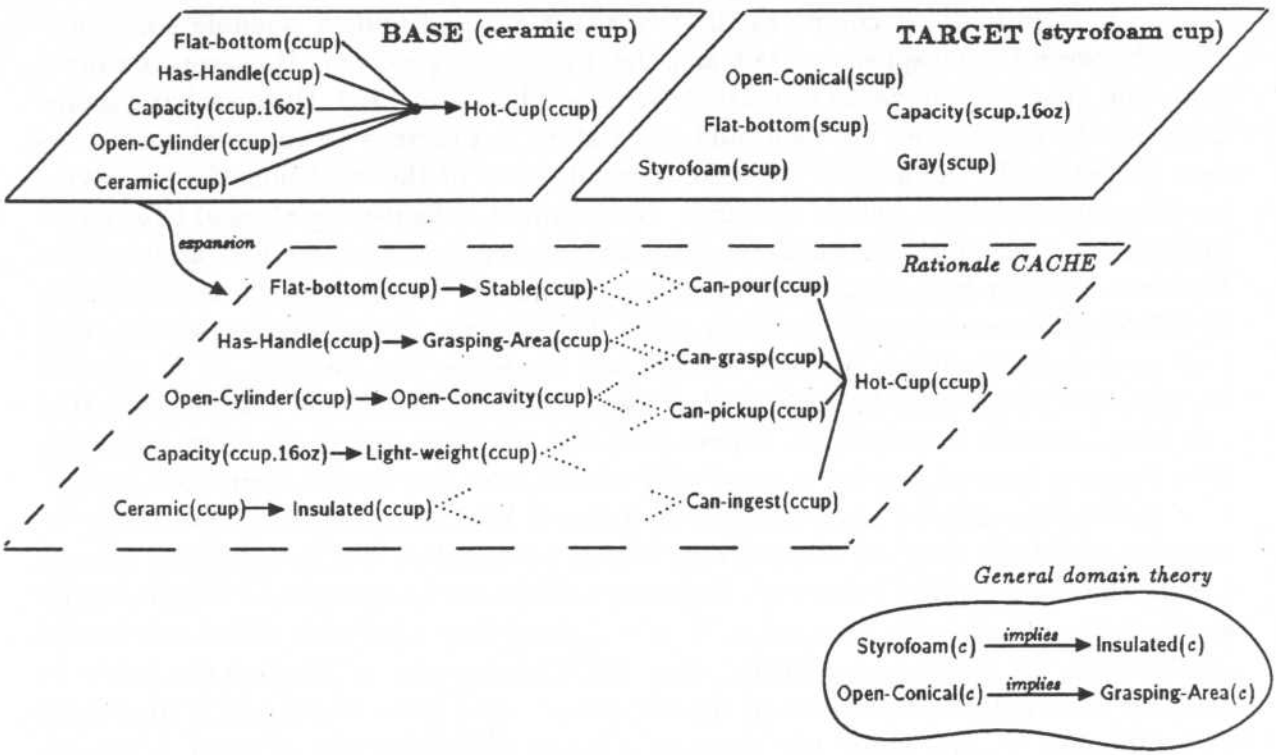

Figure 7: Categorizing a styrofoam cup by analogy to a ceramic cup.

base case's antecedents in the target description. This reveals that the styrofoam cup has the relevant *Flat-bottom* and *Capacity* properties. It also reveals that the styrofoam cup lacks the ceramic cup's other important properties: *Open-Cylinder* and *Ceramic*. For these features, PER seeks to rederive their consequents. For storage expense reasons, the actual implementation does not store these consequents explicitly. Rather, it guesses the role of an inapplicable item by retrieving a horn clause in which the item appears as an antecedent, followed by reproving the consequent. This heuristic does not guarantee that of the item's consequences, only those that play an actual role in the context of the base analogue are considered.

PER's goals are explanation-based learning and performance improvement for planning, rather than plausible analogical inference. Thus it is designed to analogically derive conclusions already entailed by existing knowledge (i.e., it assumes access to complete role information). PER picks a candidate base analogue at random from the specified class (e.g., *Hot-Cup*) and consults each antecedent in sequence. Because it does not examine the analogues' global similarity, it may waste time reproving an analogue that violates PER's *near miss* assumption.

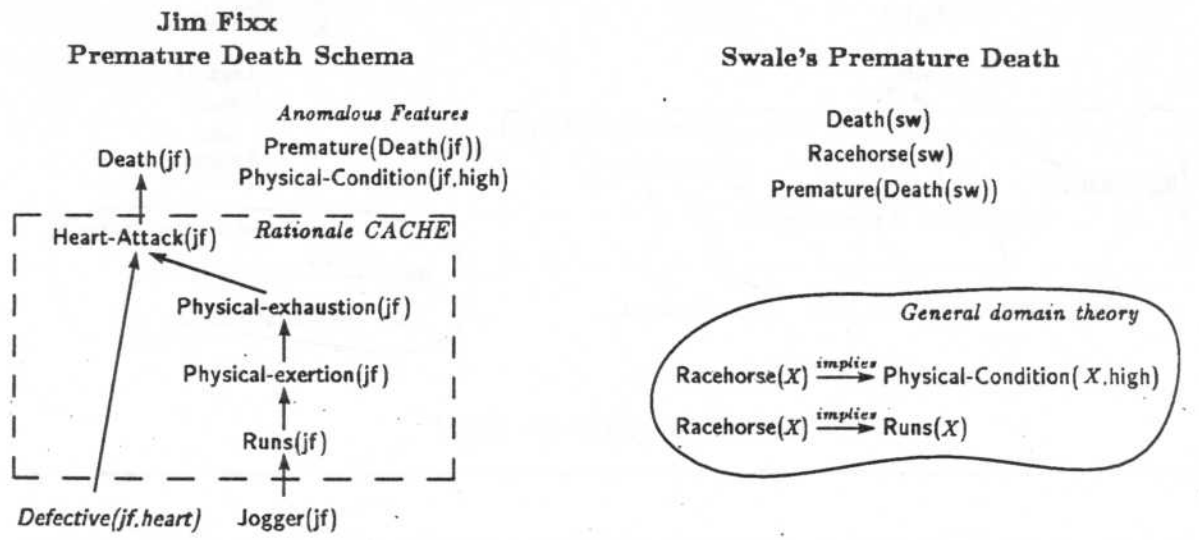

Figure 8: Why did Swale die?

SWALE [30, 29] is an explanation system that applies stored *explanation patterns* (schemas) to new situations, *tweaking* them as needed to adapt to the situation's novel aspects. For the tweaking operation, SWALE uses a set of revision rules, such as *substitute alternate theme* or *substitute related action.* These heuristics suggest ways to repair inapplicable portions of a recalled schema. For example, SWALE explains the premature death of the racehorse Swale by analogy to Jim Fixx, a famous jogger who died due to a heart defect stressed by prolonged physical exertion. Figure 8 shows a predicate calculus interpretation for SWALE's representation of this scenario. The Jim Fixx schema requires the principal actor to be a jogger, which is not true for Swale. In SWALE, such problems are treated as contradictions of the base's stated antecedents. It attempts to replace the "recreational jogger" theme (via the *substitute alternate theme* strategy) by examining each of Swale's known themes – "competed in horse races", "took performance drugs", and "ate oats". It substitutes Swale's racehorse theme for Fixx's jogger theme because only "competed in horse races" involves an action mentioned in the original explanation [29].

CSM focuses instead on the source of the problem: a dependency failure. It determines the anomalous item's role and seeks elements of the current situation that could satisfy that role's dependencies. Thus, adaptation of the Jim Fixx analogue is performed by analyzing the role of the failed jogger precondition, which is to satisfy the running action dependency, and noting that Swale's racehorse status is functionally analogous. This goal-direct behavior avoids ever considering Swale's other themes, but can require constructing complex explanations. An interesting open question is the relative efficiency of the heuristic tweaking strategies and the explicit role analysis in CSM. Alternatively,

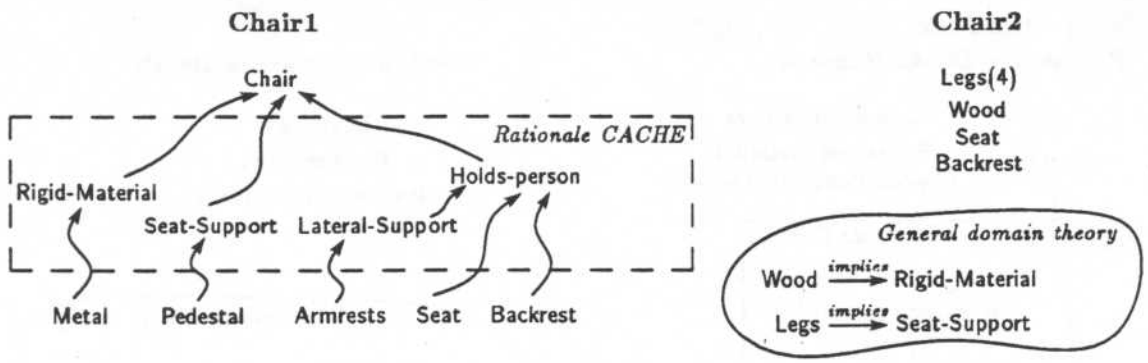

Figure 9: Is `chair-2` a chair?

role analysis could be used to form new tweaking strategies when none of the existing ones apply.

SWALE does not attempt to completely address the correspondence problem, since object level mappings are unambiguous in the small stories analyzed. Additionally, like PER, it is designed as more of a deductive rather than analogical system.

3.4 Exemplar Classification

Case-based and exemplar-based approaches interpret new cases by relating them to a store of previously classified exemplars, performing classification by finding the exemplar that best matches the new instance. PROTOS [3] implements this technique as a learning apprentice for heuristic classification and has been successfully used for diagnosis in clinical audiology. Partial, approximate explanations of each feature's relevance to the concept are stored with each category instance. These provide needed role information when classifying a new case and provide a basis for plausible generalization. Comparison of a new case to a category exemplar is performed by *knowledge-based pattern matching.* Like the approaches in the previous section, its computational performance relies on proximity of the instance descriptions. In fact, one of the central claims in PROTOS is that exemplar-based reasoning is more efficient than traditional approaches because of the large distance between the instance language and generalized concept language for many concepts.

For example, to classify `Chair-2` (described in Figure 9) as a chair, PROTOS would compare it to a known chair exemplar `Chair-1`. During this comparison, the syntactically distinct features *Legs* and *Pedestal* are considered functionally analogous because they both provide *Seat-Support.* PROTOS limits the scope of a match to the explicitly given features of the two cases. It does not include the possibility that additional features may be derived or retrieved from memory in response to the needs of the match elaboration process. Furthermore, it (and most case-based approaches) uses a feature-

vector representation (i.e., a set of attribute-value pairs). This is inadequate for complex scenarios requiring representation of the relations between objects, their features, and other relations (e.g., legal precedents and physical systems). Branting [4] addresses this limitation in PROTOS.

4 The map and analyze process

From the reviewed systems, we can identify a partial desiderata for a general analogical reasoning system. First, analogues may involve a complex pattern of relationships and events between multiple, interacting objects. This adds complexity and ambiguity to the mapping task and is why many approaches tend toward analogues consisting of unary features describing a single object. It requires the ability to use a description language in which predicates may be commutative and of arbitrary arity. Second, the analogues may be incomplete and syntactically dissimilar. To handle this, an analogy system must be able to reason about the roles and rationale of each analogue's elements, and reformulate their contents to include previously unconsidered facts and objects.

The central difficulty of elaborating an analogy is keeping the process focused. Computationally, this has two principle concerns. First, identifying matching aspects of the two analogues' initial descriptions. Second, generalizing and reformulating the descriptions via role analysis to expose further similarities and thereby adapt the base structure to the target case. To provide focus, we decompose these two concerns into a *map and analyze* cycle. First, simple pattern matching criteria are used to determine the best, initial mapping between the analogues and provide an initial estimate of their similarity. Second, this mapping is analyzed and additional information about unmatched areas is sought. When several mappings are possible, the first step prunes the search space by identifying the most promising initial mapping and providing an initial set of correspondences. Only when impasses arise, such as an expression having no apparent correspondent, is more detailed role information sought.

We begin by considering the two phases in isolation and then describe their integration.

4.1 The mapping phase

Constraints on the matching process that determine admissibility and selection generally fall into three classes [24]: *similarity criteria* restrict pairwise matching of predicates according to their similarity, *structural constraints* preserve the relational structure of the descriptions, and *contextual relevance* motivates the mapping towards solutions relevant to the needs of the performance element. The remainder of this section describes the CSM constraints and their implementation.

4.1.1 Preliminaries

Before proceeding, it will be useful to introduce some of the terminology from the SME framework for viewing theories of analogical mapping [16]. The representations given to SME are restricted to ground expressions with no quantification. For simplicity, predicate instances and entities in an analogue's description will be collectively referred to as *items*. In this framework, a distinction is made between allowable, pairwise correspondences and selected, complete mappings:

Definition 1 A *match hypothesis*, denoted $\mathrm{MH}(b_i,t_j)$, is a binary relation between a base item b_i and a target item t_j indicating that their correspondence is a candidate for inclusion in some mapping interpretation.

For example, a theory that restricted expression matches to those using the same predicate would declare a match hypothesis between all expression pairs sharing the same predicate, and no others.

Some match hypotheses may be mutually incompatible:

Definition 2 Two match hypotheses are *conflicting*, denoted $\mathrm{CONFLICTING}(MH_i, MH_j)$, if they may not appear in the same mapping interpretation.

For example, a theory specifying that no base item may match more than one target item would declare:

$$\mathrm{MH}(b,t_i) \wedge \mathrm{MH}(b,t_j) \wedge t_i \neq t_j \;\Rightarrow\; \mathrm{CONFLICTING}[\mathrm{MH}(b,t_i), \mathrm{MH}(b,t_j)]$$

Definition 3 A *global mapping*, or *gmap*, identifies a possible, complete mapping interpretation. Each gmap consists of a subset of the match hypotheses and the set of expressions the interpretation suggests is transferable to the target (called *candidate inferences*).

Throughout, a predicate-calculus-style notation will be used for expressing the rules of contextual structure-mapping. These rules operate over the base and target representations. The connectives "$\wedge$" and "$\vee$" have their standard meaning, "$\neg$" includes negation-by-failure, and "$\Rightarrow$" indicates an implication which asserts the consequent. Variables are represented by lowercase, italicized letters and are assumed universally quantified.

4.1.2 Similarity Criteria

Given a base item and a target item, what criteria are used to propose a match hypothesis between them? We start with some immediately obvious criteria and then focus on the important concept of *role* and its use in determining similarity correspondences.

First, some correspondences may simply be part of the input (e.g., given by a teacher, found during access etc.).

Rule 1 (Given Pairing) *Two items match if they are a priori designated as matching.*

GIVEN-PAIRING(b,t) $\Rightarrow$ MH(b,t)

Second, expressions using the same predicate are clearly candidates for matching.

Rule 2 (Same Functors) *Two expressions may match if they use the same predicate and these predicates are not part of a given pairing.*

FUNCTOR(b)=FUNCTOR(t) $\wedge$ $\neg$GIVEN-PAIRING(b,*anyt*) $\wedge$ $\neg$GIVEN-PAIRING(*anyb*,t)
$\Rightarrow$ MH(b,t)

Thus, given expressions $P(a, b)$ and $P(c, d)$, this rule would form a match hypothesis between the two instances of P as long as P is not part of a given pairing.

Additionally, some match hypotheses are motivated by virtue of their relationship to existing match hypotheses. For example, match hypotheses are declared for entities in corresponding argument positions of existing match hypotheses. In this manner, entities are only matched if sanctioned by their position in matching relations:

Rule 3 (Non-Commutative Corresponding Entity Forms) *Let* entity form *denote both entity tokens (e.g.,* water1*) and compound entities defined by functions (e.g., pressure*(water1)*). Two entity forms may match if they occupy the same argument position of non-commutative predicates that have already been matched and neither entity form is part of a given pairing.*

For example, given expressions $P(a, b)$ and $P(c, d)$ and a match hypothesis between the two instances of P, this rule would form match hypotheses between a and c and between b and d. An additional rule, similar to Rule 3, pairs entity forms that are arguments of commutative predicates (i.e., the "same argument position" condition is removed).

The above rules suffer from a dependence on identicality to initiate match hypotheses. Role information explicit in the representations is also used to provide a more sophisticated similarity criterion. Here, matches between expressions using different predicates may occur if the expressions are functionally analogous:

Rule 5 (Functionally Analogous) *Two expressions are considered* functionally analogous *and may match if they fill corresponding roles in the context of the structures being matched.*

$b_r \in$ ROLE(b) $\wedge$ $t_r \in$ ROLE(t) $\wedge$ MH(b_r, t_r) $\Rightarrow$ MH(b, t)

During mapping, ROLE must be discernible from direct inspection of the base and target descriptions, without inferencing or memory retrieval. For example, the above rule will form a match between expressions A and B if the base contains *Implies(A,C)* and the target contains *Implies(B,C)*.

4.1.3 Structural Constraints

Structural constraints ensure that each mapping consists of syntactically meaningful representations and control mapping redundancy with restrictions such as one-to-one.

Structurally Consistent Gentner's [21, 16] *structurally consistent* criterion requires that if expressions B_i and T_j are placed in correspondence, then their arguments must exhaustively correspond as well. CSM adopts this requirement with one important exception – when the expressions denote sets, only a non-empty correspondence between their elements is required.[9] Here, a "set" is distinguished as an unordered collection of relational structures that may be collectively referred to as a unit. They are associated to predicates taking any number of arguments. For example, relations joined by the predicate AND define a set. Other examples include the axioms of a theory, a decomposable compound object, or the relations holding over an interval of time. Intuitively, we would like to say that two sets correspond without requiring that their contents are exhaustively mapped.

If base and target propositions each contain a set as an argument, the propositions should not be prevented from matching if the sets' members cannot be exhaustively paired. For example,

$$\begin{array}{ll} \text{B:} & \mathit{Implies}[\mathit{And}(\mathcal{P}_1,\mathcal{P}_2,\mathcal{P}_3),\ \mathcal{P}_4] \\ \text{T:} & \mathit{Implies}[\mathit{And}(\mathcal{P}'_1,\mathcal{P}'_2),\ \mathcal{P}'_4] \end{array} \tag{1}$$

should match better than

$$\begin{array}{ll} \text{B:} & \mathit{Implies}[\mathit{And}(\mathcal{P}_1,\mathcal{P}_2,\mathcal{P}_3),\ \mathcal{P}_4] \\ \text{T}': & \mathcal{P}'_1,\ \mathcal{P}'_2,\ \mathcal{P}'_4 \end{array} \tag{2}$$

SME's original structurally consistent constraint [16] would score (1) and (2) equally, since the *Implies* relations of (1) would not be allowed to match. This is particularly inappropriate when matching sequential, state-based descriptions (e.g., the behavior of a system through time). The set of relations describing a pair of states often do not exhaustively match or are of different cardinality. Yet, relations over states, such as temporal orderings, are vital and must appear in the mapping.

One-To-One and Many-To-Many If a one-to-one criterion is clearly too strict, the problem remains of how to weaken the criterion while still ensuring meaningful mappings. Fortunately, the definition of *role* provides the needed specification: violations of the one-to-one restriction are motivated by a single base or target item filling multiple roles that are filled by multiple items in the other domain. Figure 10 shows the multiple dependencies motivating a many-to-one mapping in a simplified representation of the thermostats example.

[9] Unlike the other criteria of CSM, the structural consistency criteria are hardcoded in SME.

Figure 10: A many-to-one mapping motivated by role analysis.

The following three rules define *sanctioned* one-to-many mappings and enforce one-to-one mappings as the normal default by examining all cases of a single base item mapping to two target items.

Rule 6 (Direct role-filler) *A base item b, filling roles $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$, may map to multiple target items, t_1 and t_2, filling roles $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ respectively, if role $\mathcal{R}_{b1}$ corresponds to role $\mathcal{R}_{t1}$ and role $\mathcal{R}_{b2}$ corresponds to role $\mathcal{R}_{t2}$.*

$$\begin{aligned}&\text{MH}(b,t_1) \wedge \text{MH}(b,t_2) \wedge t_1 \neq t_2 \wedge \\ &r_{b1} \in \text{ROLE}(b) \wedge r_{b2} \in \text{ROLE}(b) \wedge r_{b1} \neq r_{b2} \ \wedge \ r_{t1} \in \text{ROLE}(t_1) \wedge r_{t2} \in \text{ROLE}(t_2) \wedge r_{t1} \neq r_{t2} \wedge \\ &\text{MH}(r_{b1},r_{t1}) \wedge \text{MH}(r_{b2},r_{t2}) \\ &\quad \Rightarrow \text{SANCTIONED}(b,t_1,b,t_2)\end{aligned}$$

The above rule sanctions expressions that may violate the one-to-one restriction. However, unless this sanctioning is propagated to their respective subexpressions, a one-to-one restriction will still be in effect.

Rule 7 (Role-filler sub-expressions) *Let* OCCURS-IN$(b,\ b_p)$ *denote the relationship that expression b is syntactically contained in the expression b_p. A base item may map to multiple target items if the base and target items are subexpressions of a sanctioned one-to-many mapping.*

$$\begin{aligned}&\text{MH}(b,t_1) \wedge \text{MH}(b,t_2) \wedge t_1 \neq t_2 \wedge \text{SANCTIONED}(bp,tp_1,bp,tp_2) \wedge \text{OCCURS-IN}(b,bp) \wedge \\ &\{\ [\text{OCCURS-IN}(t_1,tp_1) \wedge \text{OCCURS-IN}(t_2,tp_2)] \vee \\ &\ \ [\text{OCCURS-IN}(t_1,tp_2) \wedge \text{OCCURS-IN}(t_2,tp_1)]\ \} \\ &\quad \Rightarrow \text{SANCTIONED}(b,t_1,b,t_2)\end{aligned}$$

With sanctioned violations of one-to-one identified, we are now ready to define when two match hypotheses are conflicting. Due to the non-monotonic nature of the definition, implicit in the following rule is the assumption that all sanctioned pairings are known at the time the rule is invoked.

Rule 8 (One-To-One) *Unless explicitly sanctioned, two match hypotheses are* conflicting *if they pair the same base item to multiple target items.*

$$\begin{aligned}&\text{MH}(b,t_1) \wedge \text{MH}(b,t_2) \wedge t_1 \neq t_2 \wedge \neg\text{SANCTIONED}(b,t_1,b,t_2) \\ &\quad \Rightarrow \text{CONFLICTING}[\text{MH}(b,t_1),\text{MH}(b,t_2)]\end{aligned}$$

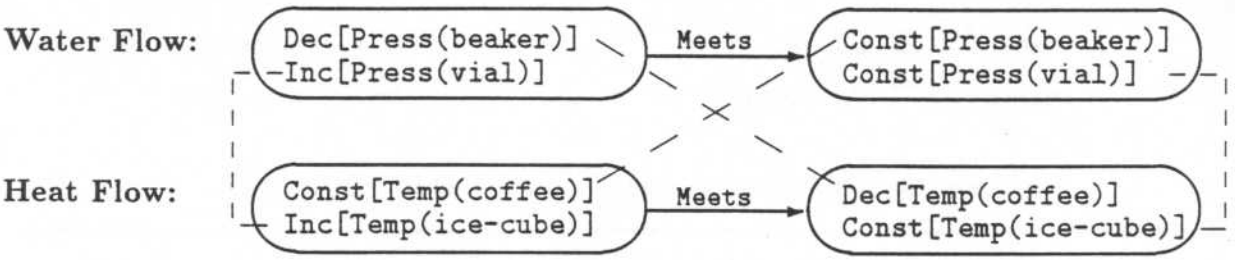

Figure 11: The structure rearrangement problem in behavioral descriptions of liquid flow and "heat flow".

A symmetric set of three rules exist for multiple base items mapping to a single target item. Together, the six rules identify sanctioned many-to-many mappings and enforce one-to-one for all other, unsanctioned cases.

Structure Rearrangement Implementation and testing of Gentner's structure-mapping theory led to the conclusion that, by being insensitive to the specific contents of the structures, purely structural constraints are insufficient to prevent anomalous mappings. The principal type of anomalous mapping is called *structure rearrangement.*

The structure rearrangement problem arises when some fundamental relationships in the representation are not preserved in the mapping. For example, consider the two-state behavioral descriptions of water flow and "heat flow" shown in Figure 11. The heat flow description has been altered to demonstrate the anomaly: the coffee temperature is constant in the first state and decreasing in the second state. When the rules of SME are applied, every item matches if the temporal *Meets* relation is excluded. That is, `Inc[Press(vial)]` maps to `Inc[Temp(ice-cube)]` while `Dec[Press(beaker)]` maps to `Dec[Temp(coffee)]`. All four water flow relations have a correspondent in the heat flow situation, but by rearranging time. While not all such rearrangements may be wrong, there is no way to make this decision without inspecting the content of the structures being manipulated.

Thus, additional constraints are supplied to capture important general representational or domain-specific knowledge about the structures being manipulated. At the current time, two very general representational constraints have been sufficient. The first preserves the cohesion and ordering of temporal states:

Rule 12 (Temporal Preservation) *Two match hypotheses are mutually inconsistent if they pair base items always co-occurring in time,* EQUALTIME(b_1,b_2)*, to target items that never overlap in time,* DISJOINTTIME(t_1,t_2)*. Likewise, two match hypotheses are mutually inconsistent if they pair target items always co-occurring in time to base items that never overlap in time.*

$$
\begin{aligned}
&\text{MH}(b_1,t_1) \wedge \text{MH}(b_2,t_2) \wedge \\
&\text{TEMPORALLY-SCOPED}(b_1) \wedge \text{TEMPORALLY-SCOPED}(b_2) \wedge \\
&\text{TEMPORALLY-SCOPED}(t_1) \wedge \text{TEMPORALLY-SCOPED}(t_2) \wedge \\
&\{ [\textsc{EqualTime}(b_1,b_2) \wedge \textsc{DisjointTime}(t_1,t_2)] \vee \\
&\quad [\textsc{DisjointTime}(b_1,b_2) \wedge \textsc{EqualTime}(t_1,t_2)]\} \\
&\qquad \Rightarrow \text{CONFLICTING}[\text{MH}(b_1,t_1),\text{MH}(b_2,t_2)]
\end{aligned}
$$

The second constraint preserves prespecified functional relationships.

Rule 13 (Functional Relationship Preservation) *Let f denote some function to be preserved (specified from domain or task). Let* $\textsc{Bound-by}(i, f)$ *denote that i is an item that should be bound by function f, that is, there exists some value v such that* $f(i)=v$ *is true. Then, two match hypotheses are mutually inconsistent if they pair an item bound by f and its value under f to another item bound by f and something other than its value under f, respectively.*

$$
\begin{aligned}
&\text{MH}(b_1,t_1) \wedge \text{MH}(b_2,t_2) \wedge \text{BOUND-BY}(b_1,f) \wedge \text{BOUND-BY}(t_1,f) \wedge \\
&\{ [f(b_1,b_2) \wedge \neg f(t_1,t_2)] \quad \vee \quad [\neg f(b_1,b_2) \wedge f(t_1,t_2)] \} \\
&\qquad \Rightarrow \text{CONFLICTING}[\text{MH}(b_1,t_1),\text{MH}(b_2,t_2)]
\end{aligned}
$$

In PHINEAS, this rule has been used to preserve the relationship between a liquid and its container. Prior to using this rule, it was possible to pair a contained liquid *Contained-Liquid*(cl_1) and its container, *Container-of*(cl_1)=can_1, to another contained liquid *Contained-Liquid*(cl_2) and a container that was not cl_2's container, such as can_3. This would arise if can_1 and can_3 shared other relevant properties, such as *Container*.

4.1.4 Selection: Combining systematicity and relevance

The previous rules produce match hypotheses and place constraints on their combination. This section presents criteria for selecting among possible mappings. It is generally agreed that in problem solving situations, the current reasoning goals have a strong influence over how an analogy develops. However, many uses of analogy lack a clear driving purpose. These include metaphorical embellishment, inductive generalization, and simply noting the similarities between two situations or stories. For these cases, Gentner's *systematicity* principle (the preference for large systems of interconnected relations) has strong psychological support [22].

Thus, a hybrid approach, influenced by both systematicity and contextual relevance, is used. In the absence of problem solving goals, systematicity serves as the sole criterion for selection. In the presence of problem solving goals, rules for supporting the current reasoning needs of the performance element fall into two categories. The *relevant inference* rule prefers gmaps that offer inferences needed by current problem solving goals. For example, if a cause for event E is sought, only gmaps offering the inference *Cause(C,E)* would be considered. This is a hard constraint, in that only interpretations containing the relevant base information are considered. The *salient features* rule supports relations that are more salient for the current reasoning task. For example, if the

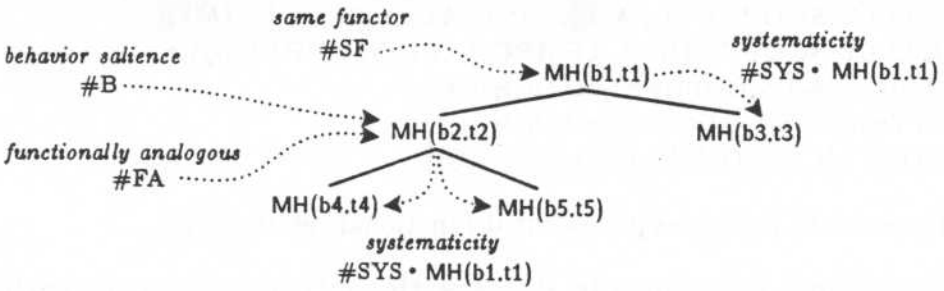

Figure 12: Match hypothesis evidence rules contribute numeric evidence scores based on factors such as systematicity and problem solving relevance.

central focus is to explain an observed behavior, then matches identifying corresponding behavior are given greater import than matches for other features.

In the implementation, SME, selection is performed by ordering gmaps according to numeric evidence scores. The rules are similar in form to the match creation and constraint rules presented above. They operate on match hypotheses, using a probabilistic combination technique to combine evidence weights supporting each match hypothesis (Figure 12. See also [16]).[10] Each evidence source, and each match hypothesis score, are in the range 0..1. A gmap's evidence is then obtained by adding the scores for each of its constituent match hypotheses. In the rules described below, each evidence weight will be given as $\#\langle parameter - name \rangle$. The numeric values of these parameters, and their performance, are then discussed in the next section.

The first three evidence rules support correspondences that are part of the original input, pair expressions having the same predicate, or pair expressions filling corresponding roles.

Rule 14 (Given Pairing Evidence) *If two items are a priori designated as matching, then supply an evidence score of* #GP *to the match.*

Rule 15 (Same Functors) *If the expressions comprising a match hypothesis use the same predicate, then supply an evidence score of* #SF *to the match.*

Rule 16 (Functionally Analogous Evidence) *If the expressions comprising a match hypothesis fill corresponding roles, then supply an evidence score of* #FA *to the match.*

Systematicity is supported by passing evidence from a match involving a relationship to the matches involving its arguments.

[10]It should be pointed out that numerical evidence is used to provide a simple way to combine local information concerning match quality. These weights have nothing to do with any probabilistic or evidential information about the base or target *per se*. Additionally, the evidence scores used here are lower than those previously described for SME_{SMT} [16]. It was found that the prior scores pushed the weights too far into the high end of the 0..1 spectrum, offering little difference between fair matches and very good matches.

Rule 17 (Systematicity) *Given* MH_1 *and* MH_2*, where* MH_2 *pairs arguments of the relations in* MH_1*, propagate* #SYS% *of* MH_1*'s total evidence score to* MH_2 *(i.e., supply an evidence score of* #SYS×MH_1 *to* MH_2*).*

In this manner, the more matched, higher-order structure that exists above a given match hypothesis, the more that hypothesis will be believed. This provides a local encoding of Gentner's systematicity principle.

Finally, contextual relevance is included. This is dependent on the performance element. In PHINEAS, the central focus is to explain an observed *behavior*. For that task, matches identifying corresponding behavior are given greater import than matches for other features.

Rule 18 (Behavior Salience) *If a match hypothesis is between two behavioral relations (e.g., Increasing, Decreasing), then supply an evidence score of* #B *to the match.*

4.1.5 Implementation

The *structure-mapping engine* (SME) [15, 16] is used to model the mapping component of contextual structure-mapping. SME is a general mapping tool which performs a set of basic mapping operations, enforces structural consistency, and formulates mappings based on user-supplied *match rules*. Match rules specify which pairwise correspondences between expressions and entities are possible, restrictions on how they may be combined, and preference criteria for scoring these combinations. The 18 rules described in the previous sections have a straight-forward translation into SME pattern-directed lisp rules, which define SME_{CSM}.[11] This section briefly describes SME_{CSM} and reviews the mapping selection criteria.

Given descriptions of base and target dgroups, SME constructs all consistent interpretations (*gmaps*) of the comparison between them. Conceptually, the algorithm is divided into four stages:

1. *Local match construction:* Given two dgroups, SME begins by finding all match hypotheses. Allowable matches are specified by *match constructor* rules (e.g., CSM rules 1-5).

2. *Gmap construction:* SME next combines local match hypotheses into maximal, consistent collections of correspondences. Consistency constraints include structural consistency and the supplied rules, which identify pairs of *conflicting* match hypotheses (e.g., CSM rules 6-13).

3. *Candidate inference construction:* The inferences suggested by each gmap are found by collecting base expressions that would fill in structure which is not in

[11] Throughout, the general program is called SME, the program running the rules of Gentner's Structure-Mapping theory is called SME_{SMT}, and the program running the rules of Contextual Structure-Mapping is called SME_{CSM}.

the gmap, provided that the base structure's subexpressions intersect at some point the base information belonging to the gmap.

4. *Match Evaluation:* An evaluation score for each match hypotheses and gmap is found by running *match evidence rules* and combining their results (e.g., CSM rules 14-18).

The specific parameter settings used in $\texttt{SME}_{CSM}$ are (the #B parameter is specific to the PHINEAS task):

Evidence Parameter	Value
#GP	0.4
#SF	0.4
#FA	0.8
#SYS	0.8
#B	0.4

Whenever numeric weights are used to influence a system's function, there is danger of (1) tailoring for particular examples and (2) sensitivity to specific values. Values for the evidence parameters in $\texttt{SME}_{CSM}$ were selected based on long experience with $\texttt{SME}_{SMT}$, empirical studies [19], and general intuition. Of course, a more formal sensitivity analysis is required. However, $\texttt{SME}_{CSM}$'s evidence parameters have not changed for several years and have been applied to dozens of examples. Both Forbus & Gentner [19] and myself have conducted empirical studies to determine $\texttt{SME}_{SMT}$'s sensitivity to the space of possible parameter settings. On extremely simple examples, it was found that simply having non-zero settings is sufficient. On more complex analogies, such as the short stories discussed in [40], performance is robust but not completely insensitive to parameter settings. Most crucial is the setting for systematicity. This must be high for $\texttt{SME}_{SMT}$ to demonstrate a marked preference for interconnected systems of relations. The findings are still preliminary. Although they should apply equally well to $\texttt{SME}_{CSM}$, studies on $\texttt{SME}_{CSM}$ have not yet begun.

4.1.6 Example: Understanding thermostats

To illustrate, Figure 13 shows a description of the thermostats originally pictured in Figure 1. The task is to use the explanation of the coil thermostat to assist in explaining the rod thermostat's functionality. To simplify the example, we will only describe the subcase in which the thermostat turns the furnace on when the temperature is less than some threshold temperature.

The initial base and target representations describe the artifacts' structural configuration. In the base case, only those aspects antecedent to the relevant explanation (i.e., is the furnace on when the temperature is below some threshold) are given. From these descriptions, $\texttt{SME}_{CSM}$ finds the following correspondences:

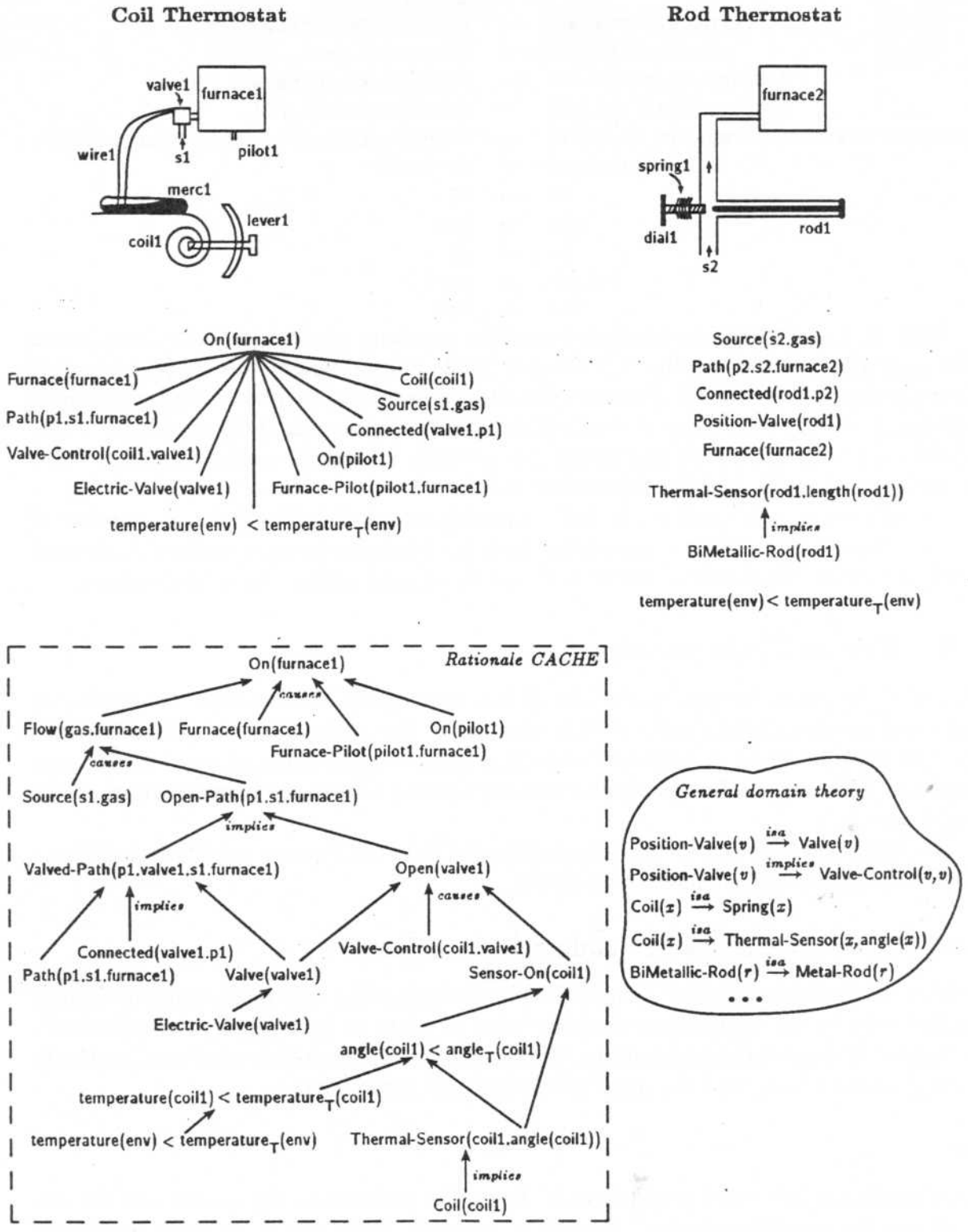

Figure 13: Descriptions of two different thermostats. How does the rod thermostat function?

Furnace(furnace1)	$\leftrightarrow$	Furnace(furnace2)
Source(s1,gas)	$\leftrightarrow$	Source(s2,gas)
Path(p1,s1,furnace1)	$\leftrightarrow$	Path(p2,s2,furnace2)
Connected(valve1,p1)	$\leftrightarrow$	Connected(rod1,p2)
temperature(env)<temperature$_T$(env)	$\leftrightarrow$	temperature(env)<temperature$_T$(env)
furnace1	$\leftrightarrow$	furnace2
s1	$\leftrightarrow$	s2
gas	$\leftrightarrow$	gas
p1	$\leftrightarrow$	p2
valve1	$\leftrightarrow$	rod1
env	$\leftrightarrow$	env

This mapping identifies relevant properties explicity present in both descriptions (SME_{CSM} rules 2-4). Due to the lack of explicit role information at this stage, no further correspondences are possible. For example, there is currently no motivation for matching Coil(coil1) with either BiMetallic-Rod(rod1) or Spring(spring1). Had additional role information been present in the initial descriptions, additional similarities could have been identified (such as coil to bimetallic rod).

At this stage, the mapping is both unambiguous and incomplete. A number of the coil thermostat's relevant properties have no (apparant) correspondent in the rod thermostat case. These initial similarities are elaborated during the analysis phase.

4.2 The analysis phase

The mapping phase determines the best, initial correspondences between two analogues and provides an initial estimate of their similarity. The analysis phase seeks additional role information about unmapped areas in a goal-directed effort to resolve mapping impasses. This exposes further similarities and thereby adapts the base structure to the target case.

We begin with the various routines needed and then use these as primitive operations in describing the analysis phase as a whole.

4.2.1 Inference engines and abductive retrieval

Portions of the analysis phase consist of seeking objects that may be assumed to occupy unfilled roles in the mapping or alternate ways to adapt or justify proposed inferences. In support of these various operations, we assume the existence of two retrieval methods – one deductive (ask) and the other abductive (abductive-ask).

ask

Some candidate inferences may represent facts that are true in the target, but are not explicit in the target description. The ask operation tests if a given expression is entailed by the current store of facts and domain theory. It's behavior is akin to Prolog or the

deductive retriever described in [8]. If the expression contains variables, then it returns a set of bindings for which the expression is true.

`abductive-ask`

The `abductive-ask` operation is identical to `ask` with one important difference – it allows abductive assumptions when deductive reasoning over existing facts is insufficient to derive a specified request pattern. When given a ground expression, `abductive-ask` tests if the expression is consistent and thus may be assumed. If the expression contains variables, then it returns a set of bindings and a set of assumptions for which the expression is true. Since skolem objects are expressed as existentially quantified variables during this operation, returned instantiations represent known objects that may consistently fill the role of a given skolem object. For example, given the request pattern

$$P(x) \wedge Q(x) \wedge R(x)$$

and assertions P(a), Q(b), and R(b), the abductive retriever would return two possibilities:

	Binding	Assumptions
1:	(x . a)	Q(a) $\wedge$ R(a)
2:	(x . b)	P(b)

The `abductive-ask` implementation used for this paper is described in [13]. It is a backward chaining, breadth-first problem solver that is responsible for maintaining the system's domain knowledge, facts, and assumptions. A branch bound *bb* (default 5) is used to limit the number of acceptable possibilities. If more than *bb* candidates are retrieved, it is assumed that not enough information exists to make a decision and the skolem object remains unknown.

Three tests are used to determine an expression's consistency: (1) the arguments are consistent with the predicate type declarations, (2) the proposition is not assumably false by closed-world assumption, and (3) the proposition's negation is not provably true. Closed-world assumptions are task-specific. For example, in the physical explanation scenarios, all spatial relationships, such as *Touching*, are assumed to be either known or false. For expressions containing variables, only argument type consistency for ground arguments is examined and the other two consistency tests are not attempted. For example, `Immersed-in(alcohol1,`x`)` is inconsistent for any instantiation of x, since `alcohol1` occupies a slot limited to solids.

4.2.2 Expression analysis and adaptation

Vocabulary used to describe the base may not be applicable to the target case, either due to alternate characteristics of the situations or vocabulary changes in crossing domains. For example, it would be better to find a known fact to fill a given role (e.g.,

Irrigation(r_t) in the coffee growing case) than to assume one imported from the base (e.g., *High-Rainfall(r_t)*). Therefore, each unmapped condition (which appears as a candidate inference from the mapping phase) is checked to see if it is true, consistent, or adaptable to fit the target environment. Three operations support this: `ask`, `abductive-ask`, and `retrieve-functional-analogue`. The ask operations were described above.

`retrieve-functional-analogue`

The `retrieve-functional-analogue` operation seeks a functionally analogous expression that may fill the role of the original base expression, and is either true or consistently assumable. A functionally analogous expression is found by determining the base expression's role in the base description and searching for a target expression that can provide the corresponding service. Cached information attached to the base description supplies the necessary information by stating what dependencies each expression satisfies. The general prerequisites the expression's base correspondent satisfied are retrieved and mapped to the target by application of the existing mapping function. Depth-bound, exhaustive backchaining is then used to locate all known target propositions deductively supporting any of these prerequisites. Each one found is added to the base and target descriptions to make their role information explicit for future mapping operations.

4.2.3 Unknown objects

When a candidate inference contains slots occupied by unknown objects (*skolem objects*), suitable target objects must be found or their existence conjectured. There are four options:

1. *General physical knowledge.* Search for a known item that may actually be the item in question. This is a component of the general abduction problem and is dealt with by the abductive retriever.

2. *Analogous conditions.* Search for a known item that satisfies constraints considered functionally analogous to those conjectured for the skolem object.

3. *Directed experimentation.* Experiments may be devised to empirically determine what the missing entity is (e.g., [39]).

4. *Hypothesize existence.* The object's existence may simply be assumed and represented by a skolem constant. If it is a known type of object, then a standard assumption mechanism as used in abduction will suffice. Otherwise, the existence of some new, hypothetical entity may be assumed, as was done when *ether* was conjectured as a medium for light waves.

Only the last option is described in this section. The first option was addressed above by `ask` and `abductive-ask`. The second option is addressed by `retrieve-functional-analogue`. The third option is not considered in this paper, but was demonstrated in [17].

`create-entity`

If a skolem object cannot be identified, a new entity token may be created and assumed to fill the role. `create-entity` makes a new entity token and then analyzes the consistency of its proposed existence.

Consistency for created entities deviates from the standard consistency operation. First, the new token is substituted into the skolem object's N conditions and the status of

$$\textit{Consistent}(C_1 \wedge \cdots \wedge C_N)$$

is determined as described above.[12] However, if the conjunction $C_1 \wedge \ldots \wedge C_N$ is inconsistent using the new entity token, then

$$(\forall x)\neg[C_1(\ldots x \ldots) \wedge \cdots \wedge C_N(\ldots x \ldots)]$$

is true, where all instances of the new entity are replaced by x. In other words, no such object could possibly exist under current beliefs. In this case, we are faced with the problem of conjecturing a new kind of entity as opposed to simply creating a new instance of a plausible entity type. In general (and for the purposes of this paper), this is grounds to reject the analogy. However, for theory formation tasks this can be grounds to form a new concept (perhaps by creating new predicates) [13, 14]. Conjectured entities have a long history in science and are often at the center of controversy in a developing theory. For example, an all-pervading *ether* was long postulated to provide a medium for the flow of light waves because other kinds of waves require a medium.

4.2.4 The map and analyze cycle

The analysis phase creates a set of usable target hypotheses from candidate inferences proposed by mapping, or rejects the analogy if this cannot be accomplished. It uses the set of operations described in the preceding sections and summarized in Table 1. The algorithm is shown in Table 2.

The process starts when a new mapping is received and begins by examining each proposed expression $\mathcal{E}$ in the set of candidate inferences. It first checks if $\mathcal{E}$ is already known using **ask**. If unknown, a functionally analogous expression that is currently believed is sought using `retrieve-functional-analogue`. If none are found and $\mathcal{E}$ is consistent (using **abductive-ask**), then $\mathcal{E}$ is assumed. Finally, if the above fails, any functionally analogous expressions that are consistent are sought. Any alternative expressions found during this process are collected and later used to augment the current base and target descriptions.

If $\mathcal{E}$ contains variables, then these operations serve to identify known objects that may be consistently substituted for the unknown object. The use of `retrieve-functional-analogue`

[12] Note that the consistency of the C_i must be determined together rather than individually, since each atomic sentence may be consistent when considered individually, but may not be consistent when considered in conjunction with the other $N-1$ assumptions. Charniak [7] makes this point as well.

Table 1: Summary of analysis phase operations.

- `ask(`*expression*`)` $\Rightarrow$ {*success* | *bindings* | *failure*}
- `abductive-ask(`*expression*`)`
  $\Rightarrow$ {*success* | $(\theta_1, \mathcal{A}_1) \ldots (\theta_N, \mathcal{A}_N)$} | *failure*
  (where θ_i is a binding and $\mathcal{A}_i$ is a set of assumptions)
- `retrieve-functional-analogue(`$P(\ldots)$`)` $\Rightarrow P'(\ldots)$
- `create-entity(`*skolem-object*`,`*conditions*`)` $\Rightarrow$ *entity-token*

in searching for unknown objects addresses an important component of the correspondence problem: how can a target correspondent for a base object be found given only the base object's stated conditions?[13] Unless these conditions have been fully mapped, there may be no target correspondent to satisfy them. This is especially true of cross-domain analogies, in which an analogous pair of objects may have no identical characteristics in common.

If all expressions are consistent and there are no unknown skolem objects remaining, the analysis phase is successfully completed. The potentially modified candidate inferences are now considered operational for the target domain (recall there may be branching at this point, producing a set of alternate target hypotheses). When this is not accomplished, there are two options. If new information was retrieved, processing returns to the mapping stage, using base and target descriptions augmented with the additional information. This may lead to an augmented or alternate mapping and possibly new points of discrepancy on which to focus the analysis process. Otherwise, the analogy is rejected.

4.2.5 Example: Understanding thermostats

The initial mapping phase found a number of relevant similarities between the two thermostats. However, five conditions and two objects from the coil thermostat had no apparant correspondents (shown before and after applying the current mapping M):

[13]Most analogy systems either create a new token and assume the proposed conditions for it [45], work on constrained within-domain analogies that eliminate the problem [6, 32], or don't examine the possibility of skolem objects produced by mapping [5, 23].

Table 2: Transfer algorithm.

Repeat Until no new information can be retrieved or there are no inconsistent expressions and no skolem objects:

1. For every expression $\mathcal{E} \in \mathcal{CI}$

   (1) `ask`($\mathcal{E}$)
   (2) else `retrieve-functional-analogue`($\mathcal{E}$) (*accept provably true alternates only*)
   (3) else `abductive-ask`($\mathcal{E}$)
   (4) else `retrieve-functional-analogue`($\mathcal{E}$) (*accept any assumable alternates*)

2. If there is no expression $\mathcal{E} \in \mathcal{CI}$ that is inconsistent and no unknown skolem objects then return target inferences $\{\mathcal{I}_{\mathcal{T}1} \ldots \mathcal{I}_{\mathcal{T}N}\}$,
   where each $\mathcal{I}_{\mathcal{T}i}$ represents a different permutation of the set of proposed modifications to $\mathcal{CI}$.

3. If (there exists an expression $\mathcal{E} \in \mathcal{CI}$ that is not consistent
   or a skolem object that cannot be identified)
   and new information has been found by `retrieve-alternate-expression`
   then invoke mapping with base and target descriptions augmented with the new information.

4. If all ground expressions are consistent but skolem objects that cannot be identified

   For every skolem object $\mathcal{S} \in$ skolem-objects($\mathcal{CI}$)
   Let $\mathcal{C}_\mathcal{S} = \{\mathcal{E}_\mathcal{S} \mid \mathcal{E}_\mathcal{S} \in \mathcal{CI}$ and $\mathcal{S} \in \mathcal{E}_\mathcal{S}\}$
   `create-entity`($\mathcal{S}$,$\mathcal{C}_\mathcal{S}$)

5. Else, reject the analogy.

$$\begin{aligned}
\texttt{Electric-Valve(valve1)} &\xrightarrow{M} \texttt{Electric-Valve(rod1)} \\
\texttt{Valve-Control(coil1,valve1)} &\xrightarrow{M} \texttt{Valve-Control(}\textit{?coil1}\texttt{,rod1)} \\
\texttt{Coil(coil1)} &\xrightarrow{M} \texttt{Coil(}\textit{?coil1}\texttt{)} \\
\texttt{Furnace-Pilot(pilot1,furnace1)} &\xrightarrow{M} \texttt{Furnace-Pilot(}\textit{?pilot1}\texttt{,furnace2)} \\
\texttt{On(pilot1)} &\xrightarrow{M} \texttt{On(}\textit{?pilot1}\texttt{)}
\end{aligned}$$

The analysis phase begins by examining each expression sequentially:

`Electric-Valve(rod1)`

1. `ask(Electric-Valve(rod1))` fails.
2. `retrieve-functional-analogue(Electric-Valve(rod1))` consults the base analogue's cache and finds that the only role of `valve1`'s `Electric-Valve` property is to support the `Valve` requirement. `Position-Valve(rod1)` satisfies this requirement for the rod thermostat case. This information is added to the analogues' descriptions:
   Base: `Implies(Electric-Valve(valve1),Valve(valve1))`
   Target: `Implies(Position-Valve(rod1),Valve(rod1))`

`Valve-Control(`*?coil*`,rod1)`

1. `ask(Valve-Control(`*?coil1*`,rod1))` succeeds (due to `Position-Valve(rod1)`) and returns {*?coil1*=`rod1`}.

`Coil(`*?coil*`)`

1. `ask(Coil(`*?coil1*`))` fails.
2. `retrieve-functional-analogue(Coil(`*?coil1*`))` finds that the only role of `coil1`'s `Coil` property is the ability to operate as a thermal sensor. `BiMetallic-rod(rod1)` provides this function in the rod thermostat context. This information is added to the analogues' descriptions:
   Base: `Implies(Coil(coil1),Thermal-Sensor(coil1,angle(coil1)))`
   Target: `Implies(BiMetallic-Rod(rod1),Thermal-Sensor(rod1,length(rod1)))`

`Furnace-Pilot(`*?pilot1*`)`

1. `ask(Furnace-Pilot(`*?pilot1*`,furnace2))` fails.
2. `retrieve-functional-analogue(Furnace-Pilot(`*?pilot1*`))` fails.
3. `abductive-ask(Furnace-Pilot(`*?pilot1*`,furnace2))` fails.

`Furnace-Pilot(`*?pilot1*`)`

1. `ask(On(`*?pilot1*`))` fails.

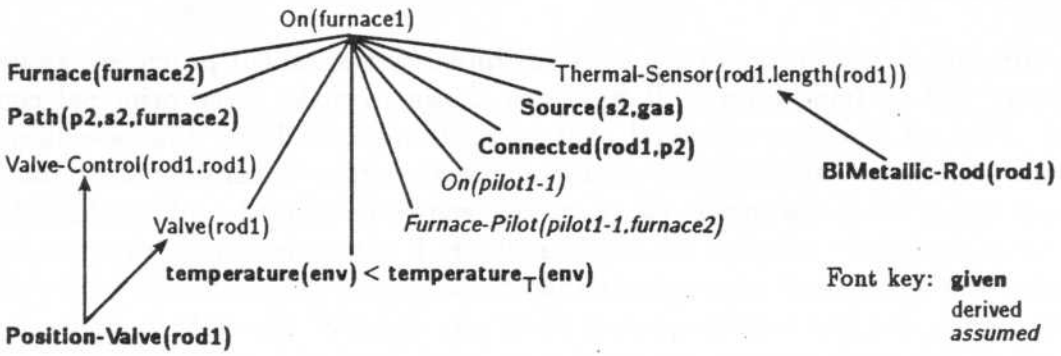

Figure 14: Final explanation of the rod thermostat's function.

2. `retrieve-functional-analogue(On(?pilot1))` fails.
3. `abductive-ask(On(?pilot1))` fails.

At this point, the analysis phase is completed. Because new role information was retrieved, the mapping is reexamined. This produces the following additional correspondences:

```
Implies(Coil(coil1),Thermal-Sensor(coil1,angle(coil1)))
      ↔ Implies(BiMetallic-Rod(rod1),Thermal-Sensor(rod1,length(rod1)))
Implies(Electric-Valve(valve1),Valve(valve1))
      ↔ Implies(Position-Valve(rod1),Valve(rod1))
valve1 ↔ rod1
coil1  ↔ rod1
```

Note that the syntactically distinct expressions `Coil(coil1)` and `BiMetallic-Rod(rod1)` are placed in correspondence due to the common role of thermal sensor (via SME_{CSM} rule 5). Additionally, `rod1` has been found to correspond to both `coil1` and `valve1` due to the rod's dual roles as sensor and valve (via SME_{CSM} rules 6-11). At this point, the only unresolved features are

`Furnace-Pilot(?pilot1,furnace2)` and `On(?pilot1)`

Because they have no correspondents and an assumption of their existence is consistent, the new entity token `pilot1-1` is created for *?pilot1*, and `Furnace-Pilot(pilot1-1,furnace2)` and `On(pilot1-1)` are assumed. The final explanation for the rod thermostat's functionality is shown in Figure 14.

5 Discussion

Elaborating the nature of role information in similarity assessment provides significant explanatory and unifying power. It has been shown to make three principal contributions. First, it provides a general characterization of similarity that specifies the conditions under which syntactically distinct expressions may be considered similar, the conditions under which the one-to-one restriction should be relaxed, and how knowledge of one situation may be adapted for use in another. It attempts to explicate the representational and algorithmic assumptions that can be made, the conditions under which they are viable, and the ramifications of their use. Second, the CSM framework has been used to unify and explain some of the intuitions behind a number of converging, independent research efforts. Its scope and utility has been extensively demonstrated on a range of tasks of varying complexity. Third, the map and analyze cycle provides a focused mechanism for computing similarity correspondences and adapting the base structure for use in the target case. The implementation allows extended representational expressiveness, is task-independent, and has been shown to have broad coverage. In addition to those described here, the implementation (which is a superset of the earlier $\texttt{SME}_{SMT}$) has been applied to a wide range of examples, including short story comparisons [40, 19], geometric figures [41], some examples from SPROUTER [26], and a dozen from PHINEAS.

The remainder of this section outlines a number of outstanding research issues.

5.1 Relationship to other learning techniques

Analogical reasoning has close ties to both inductive and explanation-based (EBL) learning. With role analysis and function substitution capabilities removed, the implementation effectively reinstantiates a stored explanation structure to the current case. This is almost equivalent to the functionality provided by EBL. Role analysis adds flexibility to the reinstantiation process by allowing sound (when role information is complete) modification of the structure. This close relationship deserves indepth attention. Which is more efficient – using traditional EBL techniques to form a range of macros covering some example space or using CSM techniques over a smaller number of adaptable macros covering the same space?

When partial or probabalistic role information is allowed, analogical reasoning lies where theory and empirical observation blend. Although some approaches make the attempt, prediction from a single case has no empirical justification and must rely on at least a weak domain theory for support. As numerous, similar cases are observed, and patterns or prototypes are identified, empirical justification becomes possible. In this manner, analogy forms a natural bridge between pure forms of inductive and analytical (explanation-based) learning. Unfortunately, it also raises a number of complex indexing, retrieval, and generalization issues that are currently not well understood.

5.2 The utility of reasoning from similarity

There are important tradeoffs that research in analogical reasoning must consider. What is it being used for? What utility does it provide? Are the costs commensurate with this utility? Analogical reasoning has two primary uses. First, it may provide faster solutions to otherwise solvable problems (e.g., planning by analogy). Second, it may provide plausible conjectures and inductive generalizations. In either case, researchers to date have focused on first understanding what it means to reason from similarities and have not yet examined the cost/benefit tradeoffs.

For what class of tasks, techniques, and analogies is analogical reasoning an effective approach? At one end of the spectrum are approaches which syntactically match lists of unary features, without further adaptation or analysis. These techniques offer efficient memory retrieval and similarity computation at the cost of reduced coverage and representational expressivity. At the other end of the spectrum lies approaches like CSM, which allows commutative, n-ary relations over multiple objects and adaptation via role analysis. These techniques offer wide coverage, including across-domain capabilities, and representational expressivity. However, they incur greater computational expense. Extensive empirical and theoretical work is needed to explore this space and understand for both simple and complex problems the effects of the various limiting assumptions that can be imposed.

One very important utility issue concerns the treatment of expressions filling corresponding roles. The technique described in Section 4 uses role analysis and the map and analyze cycle to maintain a full correspondence set while adapting the base to fit the target case. While CSM alleviates much of the brittleness associated with pattern-matching approaches to analogy, the current approach still requires that for each pair of corresponding roles, their role fillers are isomorphic (due to the structural consistency criterion). Arbitrary collections of expressions that are functionally analogous cannot be matched. For example, suppose a set of relationships between three objects serves the same role as a different set of relationships between two objects. Role analysis will successfully identify the role fillers, but the subsequent mapping phase will reject their correspondence due to their structural inconsistency. There are several responses to this issue that deserve further consideration:

- *Just fill the roles.* One could argue that role analysis should be used to simply refill the roles of unmatched base items, without attempting to explicitly name their correspondents. The techniques in [32, 37] reflect this approach. At first glance, this alternative may appear to be obviously superior and the algorithm can easily be modified to function this way (remove the return to the mapping phase in line 3 of Table 2). However, consider the following set of base expressions:

  $P(\mathrm{b}) \rightarrow Q(\mathrm{b})$
  $P(\mathrm{b}) \rightarrow R(\mathrm{b})$
  $P(\mathrm{b}) \rightarrow S(\mathrm{b})$

Upon finding $P'(t) \rightarrow Q(t)$ in the target, which suggests that $P(b)$ corresponds to $P'(t)$, this approach would not know to try $P'(t)$ as the first candidate when attempting to justify $R(t)$ and $S(t)$. Explicitly noting correspondences uncovered during role analysis may provide focus for subsequent problem solving.

- *Record all functional correspondences.* In addition to its focusing potential, some tasks may require explicitly noting all correspondences (e.g., forming abstractions, comparing and contrasting legal cases [1], or evaluation of competing designs). However, if matches between syntactically disparate role fillers must be recorded, this approach raises serious issues for identifying object and relational correspondences. For example, consider matching

  $P(b) \rightarrow Q(b)$

  with

  $R(t_1,t_1) \wedge S(t_2) \rightarrow Q(t_1)$

  Should we just place the antecedents in sets and say that $\{P(b)\}$ corresponds to $\{R(t_1,t_1), S(t_2)\}$, without further decomposition? What impact does this have on structural consistency? What would such a correspondence mean?

- *Take the middle ground.* A final approach of course is simply to refill the roles of unmatched base items and record any correspondences that can be easily identified.

5.3 Research issues for analogical mapping

5.3.1 The plausibility problem

What is the underlying basis for analogical inference that makes conclusions plausible? As discussed in Section 1.3, several factors contribute to evaluating the plausibility of analogical inferences, including logical or statistical justification [9, 10, 35], consistency [28], and empirical utility [12, 13, 23]. The general condition that there be a predictive correlation between mapped features and inferred features appears fundamental. However, the plausibility problem remains as a critical open research issue.

One dominate point of convergence has been the centrality of consistency in guiding and evaluating analogy production [28, 23, 32, 24]. Many go beyond consistency to require that analogy produces deductively sound inferences [6, 31, 10]. While consistency is an important component of all forms of reasoning, it should not be afforded too much import. Limiting analogy to strict consistency requires that analogy be a monotonic process. However, analogy often causes the questioning of beliefs and may lead to a complete change in world view. Thus, a weaker form of consistency is needed – one which takes into account the cost of overthrowing or revising prior beliefs for the benefits of a more coherent belief state. Work on explanatory coherence [38, 42] may be viewed as a step in that direction. Additional factors such as plausibility and specificity in accounting for the phenomenon are required as well.

5.3.2 The multiple instances problem

An implicit assumption in analogy research is that each analogue description depicts a single (sometimes complex) concept. For many domains and application tasks, this assumption may not be satisfied. For example, three containers connected in series (e.g., `can1` to `can2` by `pipe12` and `can2` to `can3` by `pipe23`) can produce two simultaneous liquid flow processes. Using an analogue describing the flow of liquid between two connected containers to explain this scenario would produce two different analogies, one with the `can2` to `can1` flow and the other with the `can2` to `can3` flow. This produces two independent initial explanations. Existing analogy systems do not have the ability to recognize that a single phenomenon is simply occuring twice.

5.3.3 Role analysis

More work is needed to formally capture and increase understanding of role information and its analysis. This leads to consideration of some related subproblems:

Compiled Knowledge Problem. AI systems tend to use compiled knowledge, in which intermediate reasoning steps or detailed models are absent to promote efficiency of use. This runs counter to the need in analogy to understand the underlying rationale behind that which is being adapted (i.e., its role in the analogue). For example, knowing why a prior decision was made is necessary for satisfying the intent of a decision without necessarily adhering to the same decision. More work is needed to specify what knowledge is needed, what knowledge should be cached, how that knowledge is to be retrieved, and how that knowledge is to be used in the analogy process.

Learning role information. This paper has claimed that analogy requires role information to at least plausibly suggest the relevance and interrelatedness of each analogue's features. The ability to learn this relevance information is of fundamental importance. One approach is to use a developing analogy to motivate specific questions about the world and use directed experimentation to answer them and ascertain the requisite relevance information. This was demonstrated by the merger of `PHINEAS` and `ADEPT` [17]. A second approach is to again use a developing analogy to motivate specific questions, but place the system in a learning apprentice setting and obtain requisite relevance information from the user. This is the approach taken in `PROTOS` [3]. Finally, a third approach would be to use inductive methods to indicate which factors are relevant to the concept under study.

6 Acknowledgements

This work has benefited from guidance, suggestions, and generally enlightening conversations with Danny Bobrow, John Collins, Ken Forbus, Dedre Gentner, Pat Langley,

Mark Shirley, Janice Skorstad, VS Subrahmanian, and Brian Williams. Danny Bobrow and Mark Shirley were instrumental to the development of these concepts beyond their specific instantiation in PHINEAS.

Portions of this work were conducted as part of the author's doctoral disseration at the University of Illinois Department of Computer Science. Support was provided through an IBM Graduate Fellowship and the Office of Naval Research, Contract No. N00014-85-K-0559.

References

[1] Ashley, K. D and Rissland, E. L. Compare and contrast, A test of expertise. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 273–278, Seattle, WA, July 1987. Morgan Kaufmann.

[2] Baker, M, Burstein, M. H, and Collins, A. Implementing a model of human plausible reasoning. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 185–188, Milan, Italy, August 1988. Morgan Kaufmann.

[3] Bareiss, E. R, Porter, B. W, and Wier, C. C. Protos: An exemplar-based learning apprentice. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 12–23, Irvine, CA, June 1987. Morgan Kaufmann.

[4] Branting, L. K. Integrating generalizations with exemplar-based reasoning. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Ann Arbor, MI, August 1989. LEA.

[5] Burstein, M. Concept formation by incremental analogical reasoning and debugging. In *Proceedings of the Second International Workshop on Machine Learning*, Monticello, IL, June 1983. (Revised version appears in R.S. Michalski, J. Carbonell, T. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach, Volume II*, Morgan Kaufmann, 1986).

[6] Carbonell, J. G. Derivational analogy in problem solving and knowledge acquisition. In *Proceedings of the Second International Workshop on Machine Learning*, Monticello, IL, June 1983. A revised version appears in *Machine Learning: An Artificial Approach Vol. II*, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell (Eds.), Morgan Kaufman, 1986.

[7] Charniak, E. Motivation analysis, abductive unification, and nonmonotonic equality. *Artificial Intelligence*, 34(3):275–295, 1988.

[8] Charniak, E, Riesbeck, C. K, and McDermott, D. V. *Artificial Intelligence Programming.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1980.

[9] Clark, P. Representing arguments as background knowledge for the justification of case-based inferences. In *Proceedings of the AAAI-88 Workshop on Case-Based Reasoning*, pages 24–29, August 1988.

[10] Davies, T. R and Russell, S. J. A logical approach to reasoning by analogy. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 264–270, Milan, Italy, August 1987. Morgan Kaufmann.

[11] DeJong, G and Mooney, R. Explanation-based learning: An alternative view. *Machine Learning*, 1(2), 1986.

[12] Falkenhainer, B. An examination of the third stage in the analogy process: Verification-Based analogical learning. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, August 1987. Morgan Kaufmann.

[13] Falkenhainer, B. *Learning from Physical Analogies: A Study in Analogy and the Explanation Process.* PhD thesis, University of Illinois at Urbana-Champaign, 1988. (Technical Report UIUCDCS-R-88-1479).

[14] Falkenhainer, B. A unified approach to explanation and theory formation. In Shrager, J and Langley, P, editors, *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufmann, San Mateo, CA, 1990. Also appears in *Readings in Machine Learning*, Shavlik & Dietterich (Eds.), 1990.

[15] Falkenhainer, B, Forbus, K. D, and Gentner, D. The structure-mapping engine. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 272–277, Philadelphia, PA, August 1986. Morgan Kaufmann.

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

[17] Falkenhainer, B and Rajamoney, S. The interdependencies of theory formation, revision, and experimentation. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 353–366, Ann Arbor, MI, June 1988.

[18] Forbus, K. D. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[19] Forbus, K. D and Gentner, D. Structural evaluation of analogies: What counts? In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, pages 341–348, Hillsdale, NJ, August 1989. Lawrence Erlbaum Associates.

[20] Gentner, D. The structure of analogical models in science. Technical Report BBN Technical Report No. 4451, Bolt Beranek and Newman Inc., Cambridge, MA., 1980.

[21] Gentner, D. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170, April-June 1983.

[22] Gentner, D. Mechanisms of analogical learning. In Vosniadou, S and Ortony, A, editors, *Similarity and Analogical Reasoning.* Cambridge University Press, London, 1988.

[23] Greiner, R. Learning by understanding analogies. *Artificial Intelligence*, 35(1):81–125, 1988.

[24] Hall, R. P. Computational approaches to analogical reasoning: A comparative analysis. *Artificial Intelligence*, 39(1):39–120, May 1989.

[25] Hammond, K. J. CHEF: A model of case-based planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 267–271, Philadelphia, PA, August 1986. Morgan Kaufmann.

[26] Hayes-Roth, F and McDermott, J. An interference matching technique for inducing abstractions. *Communications of ACM*, 21(5):401–411, 1978.

[27] Holyoak, K. J and Thagard, P. Analogical mapping by constraint satisfaction. *Cognitive Science*, 13(3):295–355, 1989.

[28] Indurkhya, B. Approximate semantic transference: A computational theory of metaphors and analogies. *Cognitive Science*, 11:445–480, 1987.

[29] Kass, A. Adaptation-based explanation: Extending script/frame theory to handle novel input. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 141–147, Detroit, MI, August 1989. Morgan Kaufmann.

[30] Kass, A, Leake, D, and Owens, C. SWALE, A program that explains. In Schank, R, editor, *Explanation Patterns: Understanding Mechanically and Creatively.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[31] Kedar-Cabelli, S. T. Purpose-directed analogy. In *Proceedings of the Seventh Meeting of the Cognitive Science Society*, August 1985.

[32] Kedar-Cabelli, S. T. *Formulating Concepts and Analogies According to Purpose.* PhD thesis, Rutgers University, May 1988. (Technical Report ML-TR-26).

[33] Kolodner, J, Simpson, R. L, and Sycara-Cyranski, K. A process model of case-based reasoning in problem solving. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 284–290. Morgan Kaufmann, August 1985.

[34] Koton, P. Reasoning about evidence in causal explanations. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 256–261, St. Paul, MN, August 1988.

[35] Loui, R. P. Analogical reasoning, defeasible reasoning, and the reference class. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 256–265, Toronto, CA, May 1989. Morgan Kaufmann.

[36] Mitchell, T, Keller, R, and Kedar-Cabelli, S. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1), 1986.

[37] Mostow, J. Design by derivational analogy: Issues in the automated replay of design plans. *Artificial Intelligence*, 40:119–184, 1989.

[38] Ng, H. T and Mooney, R. J. On the role of coherence in abductive explanation. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 337–342, Boston, MA, July 1990. AAAI Press.

[39] Rajamoney, S. Experimentation-based theory revision. In *Proceedings of the 1988 AAAI Spring Symposium Series: EBL*, Stanford, CA, March 1988.

[40] Skorstad, J, Falkenhainer, B, and Gentner, D. Analogical processing: A simulation and empirical corroboration. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, Seattle, WA, July 1987. Morgan Kaufmann.

[41] Skorstad, J, Gentner, D, and Medin, D. Abstraction processes during concept learning: A structural view. In *Proceedings of the Tenth Meeting of the Cognitive Science Society*, Montreal, August 1988.

[42] Thagard, P. Explanatory coherence. *Behavioral and Brain Sciences*, 12:435–502, 1989.

[43] Wellsch, K and Jones, M. Computational analogy. In *Proceedings of the Seventh European Conference on Artificial Intelligence (ECAI)*, pages 153–162, July 1986.

[44] Winston, P. H. Learning and reasoning by analogy. *Communications of ACM*, 23(12), December 1980.

[45] Winston, P. H. Learning new principles from precedents and exercises. *Artificial Intelligence*, 19:321–350, 1982.