Proceedings of the AAAI Spring Symposium on Representing Mental States and Mechanisms, Stanford, CA, March 1995.

Modeling Case-based Planning for Repairing Reasoning Failures

Susan Fox **David B. Leake**
Computer Science Department
Indiana University
Bloomington, IN 47405
{sfox,leake}@cs.indiana.edu

Abstract

One application of models of reasoning behavior is to allow a reasoner to introspectively detect and repair failures of its own reasoning process. We address the issues of the transferability of such models versus the specificity of the knowledge in them, the kinds of knowledge needed for self-modeling and how that knowledge is structured, and the evaluation of introspective reasoning systems. We present the ROBBIE system which implements a model of its planning processes to improve the planner in response to reasoning failures. We show how ROBBIE's hierarchical model balances model generality with access to implementation-specific details, and discuss the qualitative and quantitative measures we have used for evaluating its introspective component.

Introduction

Many motivations underlie current interest in introspective reasoning and learning. From a functional perspective, introspective reasoning has the potential benefit of allowing the reasoner to refine its own reasoning methods, expanding its capabilities over time and adapting its reasoning to respond effectively to novel circumstances. In complex domains it is difficult or impossible to predict all the knowledge and reasoning methods the system will need ahead of time. A system which can learn new knowledge *and* new reasoning methods should be able to perform better under those circumstances. From a more general perspective, development of a model for this task will help us to understand and evaluate reasoning behavior and the knowledge needed to capture it.

In order to learn about its reasoning methods, a system must be able to detect opportunities to learn, which are defined in our system by places where expectations about ideal system performance fail (Leake, 1992; Krulwich, Birnbaum, & Collins, 1992; Hammond, 1989; Ram, 1989; Schank, 1986; Riesbeck, 1981). When actual performance differs from expected ideal performance, the system learns by assigning blame for the failure, and repairing the flaw in the underlying system. All these tasks require knowledge about how the system reasons, and what the expected results of that reasoning are. There are several different recent approaches to the task of introspective reasoning: RAPTER (Freed & Collins, 1994a, 1994b) uses expectations about a reactive planning task to diagnose and repair failures, Meta-AQUA (Ram & Cox, 1994) maintains a set of templates for reasoning failures with applicable repairs to apply to failed reasoning traces, Autognostic (Stroulia & Goel, 1994) uses an Structure-Behavior-Function model of its own reasoning to find learning opportunities, and IULIAN (Oehlmann, Edwards, & Sleeman, 1994, 1995) uses questions about its own reasoning and knowledge to re-index its memory and to regulate its processing. Our approach, ROBBIE[1] (Fox & Leake, 1994), models the desired behavior of its underlying case-based planning component as a set of expectations about the behavior of the system during the planning process. ROBBIE monitors the reasoning of its underlying system, comparing its performance to a model of the "ideal" performance of the case-based reasoning process, as first proposed by Birnbaum et al. (1991). The model contains expectations about each portion of the system's reasoning processes. These expectations, assertions that would hold for an ideal CBR system, are organized by the component of the system they refer to, their level of specificity, and their relations to other expectations. The questions of what expectations are required, at what levels of abstraction, and how they relate to each other lie at the heart of this work.

In this paper we focus on a few issues of importance to systems which use introspective reasoning for self-improvement. In particular we consider the tradeoff between creating a general, transferable model and creating a model with sufficient detail to guide precise diagnosis and repairs, and we consider the issue of evaluating introspective learning as a methodology and in terms of specific uses.

Generality vs. Specificity: In order to facilitate the application of a self-modeling framework to many different systems, we must keep the model as general as possible and use mechanisms independent of both the implemented system and the particular task. At the same time detailed descriptions of the underlying mechanisms and domain are needed in order for the self-model to determine concrete repairs. In ROBBIE, we propose an approach to introspective learning that strikes a balance between the desired generality and the needed specificity, and which has other benefits of its own in simplifying access to the model.

The mechanisms in ROBBIE which manipulate its introspective model are independent of ROBBIE's domain and underlying system, providing a few simple means of communication between introspective reasoner and underlying system. The vocabulary in which the model is represented is designed to describe data and reasoning tasks without being specific to a particular implementation. The model structure preserves as much generality as possible by maintaining a hierarchy of assertions (expectations) which keep task- and implementation-specific details separate from generalities that might be more transferable to other tasks and domains (Fox & Leake, 1994).

[1] Re-Organization of Behavior By Introspective Evaluation

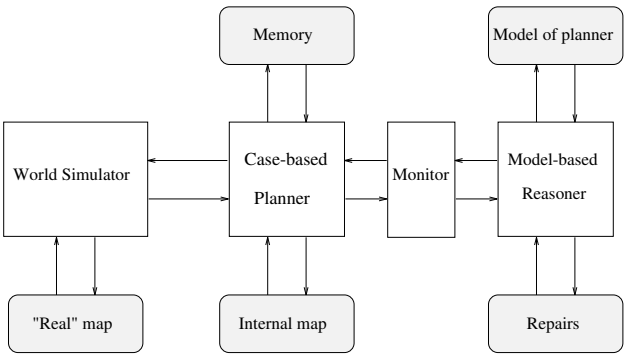

Figure 1: ROBBIE Architecture

Evaluating the method: Evaluation of AI systems is important to verify that the claims made about their performance actually hold. Up to this point little concrete evaluation has been attempted for introspective reasoning systems; we will discuss possible means for evaluating such systems and describe how we have begun to evaluate ROBBIE.

In order to fully analyze ROBBIE's performance, we must develop criteria for judging how good or "useful" its method is; we must justify the effort expended both in terms of what we may learn about modeling mental states and in terms of the tangible benefits of designing such a system.

By analyzing ROBBIE's approach, we can learn something of the knowledge needs of systems for doing introspective diagnosis and repair, and of how that knowledge should be structured. For example, expectations at multiple levels of abstraction seem to make the modeling as well as the transferring of goals more tractable. Making fine discriminations among the kinds of relationships between expectations seems to improve the focus of assigning blame when a failure does occur.

One practical justification for using introspective reasoning is the potential for improved performance; to support such a claim we must determine, quantitatively and qualitatively, to what extent performance has improved. Potential evaluation methods should provide some measure of the magnitude of improvement introspective reasoning produces: one possible evaluation method is to compare the performance of the bottom-level system alone with that of the system as a whole. In addition, we should define more qualitative methods, such as learning the "right" new reasoning, or producing "better" output results.

We first describe the ROBBIE system in detail and present an example of the sort of introspective learning ROBBIE performs. Then we will consider the issues described above to see how ROBBIE fits in and what we can conclude.

The ROBBIE system

The ROBBIE system is, at the most basic task level, a planning system, which interacts with a user and a simulated world to generate and execute plans for that world. That "performance" task is performed by a case-based planner (Hammond, 1989; Alterman, 1986; Kolodner, 1993), combined with a simple reactive-style execution system (Firby, 1989). Overarching the performance task is the task of learning introspectively about the planning and execution process itself, which is done using model-based reasoning about the system's own reasoning process (Birnbaum et al., 1991; Collins, Birnbaum, Krulwich, & Freed, 1993; Birnbaum, Collins, Freed, & Krulwich, 1990). This higher-level task is performed by a separate component which interacts with the planner (see Figure 1).

Presented with a starting location (usually the current location of the simulated robot) and a goal location to reach, ROBBIE's case-based component retrieves the most similar matching solution in memory. Similarity is initially judged by a naive method comparing the geographic "closeness" of the starting and goal locations in the current situation to those in the solutions in memory. ROBBIE can learn new features to use in assessing similarity. The solution retrieved from memory is adapted by trying to map the actual starting and ending locations onto the retrieved ones. The resulting plan is executed by the reactive planning component, taking each high-level plan step as a goal to be reached. This execution provides an evaluation of the quality of the adapted plan.

During the plan generation and execution process, the introspective component monitors the reasoning of the case-based and reactive components for discrepancies between its expectations and the actual results. ROBBIE uses a model of the underlying planning process to provide expectations about its performance. The model is a structured set of assertions about the ideal behavior of the case-based planner (Birnbaum et al., 1991; Fox & Leake, 1994). During the monitoring process, only those assertions relevant to the current portion of the reasoning task need be considered. In diagnosing a discovered failure, the entire model may be reconsidered as a problem might not be discovered until well after it was introduced (i.e., retrieval of a bad case might not produce an explicit failure until plan execution).

The failures ROBBIE may detect include both catastrophes in which the planner incorrectly solves a problem or cannot reach a solution, and hidden failures which involve inefficient processing or successful but non-optimal solutions. For example, ROBBIE expects that it will know and use all the relevant features of a problem to retrieve the best old solution. This assertion could be violated, yet a solution still be possible from the less-than-optimal retrieved case.

When a discrepancy is discovered, the network of related assertions is reconsidered, drawing from a trace of the reasoning so far and those portions of the model reachable from the original failed assertion. Through this process the system will determine the root cause and possible repair for the noticed failure. For the failure above (that it will know and use all relevant features), ROBBIE might discover in storing the solution gained by a poorly retrieved case that the solution retrieved was *not* the best one. The introspective reasoner can work back from that noticed failure to the deeper cause: the lack of a relevant feature. ROBBIE can alter the features used in retrieval to include one that would have distinguished the "real" best solution. The example below addresses this problem in more detail.

The planner may be suspended while a repair is found and implemented, or it may be permitted to continue until more information becomes available to the introspective reasoner. After a repair has been implemented, the planner may continue from the point where a problem was observed or may be reset to a prior point in the reasoning task from which the

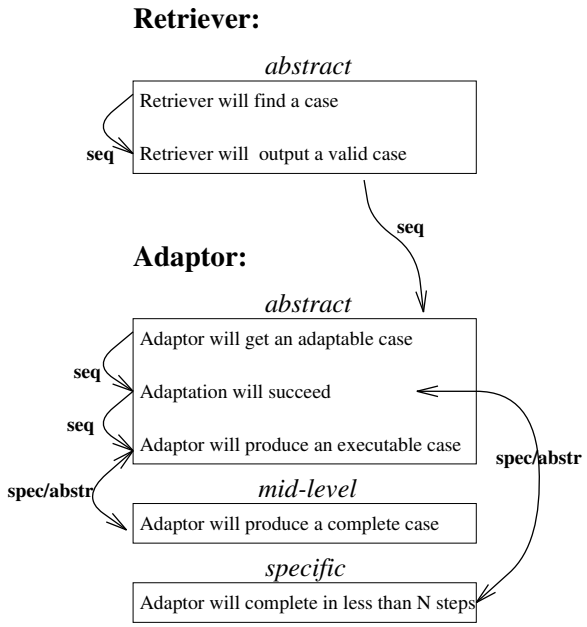

Figure 2: Sample Assertions

system can proceed normally.

ROBBIE's self-model

The introspective reasoning model is used to monitor the system's reasoning processes, and to diagnose and repair failures that occur when the assertions of ideal reasoning performance fail to be true of the actual reasoning performance. The assertions describe expectations about the reasoning processes for each component of the planning system; Figure 2 shows a portion of the current model for ROBBIE, with assertions described in English. In this section we will describe what assertions the model contains, how they are structured, what that means for the assertions in Figure 2, and the benefits gained by a hierarchical model.

Assertions in the model

The model must provide expectations for the reasoning processes of each component of the planner. The case-based planning system consists of components which perform specific parts of the CBR task: Anticipator, Retriever, Adaptor, Executor, and Storer. The Anticipator takes an initial problem description and creates an index to compare to the cases in memory. The Retriever uses that index to select the most similar solution in memory, the Adaptor changes the old solution to match the new problem, the Executor evaluates the solution by executing it, and the Storer adds the new solution to memory for future use.

The assertions in the model describe the components at different levels of specificity. At the abstract level are assertions much like the description given above. High-level assertions provide a trace of the overall flow of control and information through the planner, without using any details specific to ROBBIE. At lower levels, assertions refer to specific aspects of ROBBIE's implementation: the algorithms used for doing retrieval, adaptation, execution, and so forth. Lower-level assertions often have repairs associated with them, because they can refer to actual parts of ROBBIE which can be altered.

Several components of the planner are implemented as case-based systems themselves, sharing the same memory and retrieval mechanisms as the planner as a whole. For example, anticipation is viewed as a process of selecting and applying cases which specify features to be added to the problem description. Because of the re-use of the case-based mechanisms for more than one purpose, the details of the model are simplified for those case-based components; the model of CBR as a whole provides expectations for each of them, as well as for the planner.

Structure of the model

The assertions are structured by the component to which an assertion refers, the level of specificity of the assertion, and by connections to other related assertions. Dividing assertions into groups by their components facilitates monitoring the reasoning processes for deviations; the only assertions which must be monitored refer to the current component of processing. Assertions which belong to a particular component are also likely to be closely related to each other, as well.

Assertions are arranged hierarchically depending on how specific they are to ROBBIE's implementation. A separation by hierarchy simplifies the task of updating the model when things change, and transferring portions of the model to new underlying systems. In addition, it separates different ways of thinking about the reasoning task: the abstract levels link components together and describe how information and control passes between them, low-level assertions describe portions of particular components and the specific information needs and algorithms for them.

Each assertion is linked to the other assertions which are related to it. These links guide the introspective reasoner in explaining and repairing a detected failure by focusing on the most fruitful portions of the model. There are four kinds of links, which the introspective reasoner treats differently during the search for the deep cause of a failure: an abstraction link connects a low-level assertion to its high-level counterpart, a specification link symmetric to the abstraction link, a sequence link connects two assertions (at the same level of specificity) when one assertion refers to an earlier part of the reasoning process, and a co-occurs link connects two assertions which tend to fail or succeed together. These classes of links between assertions are preliminary; we expect to refine the classes as the model is completed.

Sample of the model

Figure 2 represents a portion of the model ROBBIE uses, showing a subset of the assertions for two components of the case-based planner. Assertions are grouped, first by component, and then by level of specificity. The number of levels depends on the component in question; this figure shows three levels for the Adaptor: abstract, mid-level, and specific. Assertions which are grouped by specificity and component are considered together during the monitoring process.

Assertions are connected together by several different kinds of links; three appear in Figure 2: "seq," "spec," and "abstr." The "seq" links encode the order in which event occur in the underlying system, "spec" and "abstr" links are symmetric and link assertions at one level to corresponding specifica-

tions or abstractions at another level. Assertions are written in English for convenience, the actual assertions use a limited vocabulary in predicate calculus.

The first component in Figure 2 is the Retriever, for which only two abstract assertions appear. In the complete model, these assertions have links to other abstract and specific assertions omitted here to simplify the example. The first assertion states that the retriever will always find some matching case. The "seq" link indicates that the next assertion comes later in the retrieval process: It states that the final result of retrieval will be the right kind of case. A memory might contains different kinds of memory structures (as ROBBIE's does), this asserts that the Retriever will find a plan, and not (for instance) an adaptation strategy, if it is looking for a plan.

Sequence links often connect assertions from two different components at the abstract level. The next assertion in sequence is an Adaptor assertion, that the Adaptor will be given an adaptable case ("adaptable" would be defined by specific assertions not included here). It is followed by an assertion stating that adaptation will succeed in producing *some* answer in a limited amount of time. This assertion is linked to a specification that describes, in details specific to ROBBIE's implementation, exactly how to judge "success" of the adaptor. The last abstract assertion is, as for the Retriever, concerned with the correct output for the component; it is linked to a mid-level assertion which defines "executable" in terms of "complete", and would have more specific assertions below it.

Benefits of a hierarchical model

A hierarchical model such as ROBBIE uses provides two advantages over an approach using just general or specific expectations. First, for knowledge re-use, we can encapsulate those parts of the knowledge which *would* apply across different systems and keep that part of the model's knowledge when doing the transfer. Equally important, however, is the value of both kinds of expectations in monitoring and repairing the underlying system. High-level assertions provide an overview of the planner's process, and allow us to connect the functioning of one component with another at the "right" level of abstraction: we don't need to trace each specific step from storing the solution back to creating the index; the abstract assertions provide access to other components at a general flow-of-control level of description. In searching for the root cause of a failure, we can use the high-level assertions to select appropriate components to consider and, from there, the appropriate specific assertions for that component. Without the lower-level assertions which describe the actual processing of this systems, it becomes nearly impossible to detect failures or to specify good repairs for them. We therefore must design a model which incorporates both levels of description; ROBBIE's hierarchical and component-oriented model is one such design.

There are still many unanswered or incompletely-answered questions about ROBBIE's approach. ROBBIE's current model is incomplete, incorporating a fraction of the assertions we expect to need, and having very few repairs at its disposal. Our immediate task is to expand the model and repairs: to do this we must determine to a finer degree what knowledge is required. We must consider how many levels of abstraction in the model hierarchy are useful for ROBBIE. We must also catalog more completely the kinds of links between assertions, as we see what effect the current divisions have on the model's processing. Ideally we would test the model structure under fire by using it to implement introspective reasoning for a different underlying system.

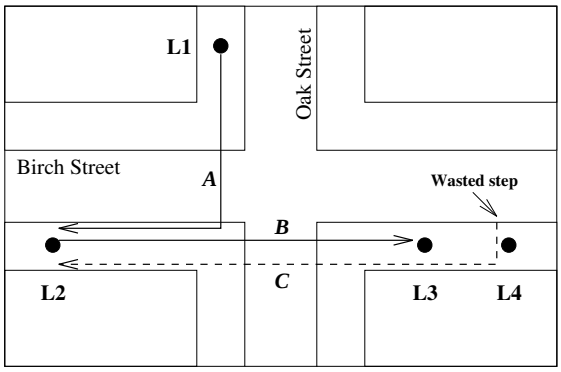

Figure 3: Map of simulated world

Plan A:

- Turn south
- Move south to south side of Birch
- Turn west
- Move west to L2

Plan B:

- Turn east
- Move east to L3

Figure 4: Plans in memory

Example: learning new index features

To make the discussion more concrete, let us consider a case in which ROBBIE alters the set of features used to index its memory. ROBBIE's underlying task is to create and execute plans for navigating city streets in a simulated world as a pedestrian. The system has access to previous routes it has taken and to a map of the world which does not include dynamically changing details. Such details at the present time include traffic lights against which the system must not cross and which break down, and street-closings. The case-based process must measure the similarity between the goal index and the indices of cases in memory to select the case which is easiest to adapt into a new solution; ROBBIE originally selects cases based on how similar the starting and ending locations are to those in memory. Such an index, while it seems an obvious approach, is not sufficient, as the following example will make clear.

Figure 3 shows a portion of the world map relevant to this problem. ROBBIE has in memory plan A, which describes how to travel from location L1 to location L2, and plan B, which describes how to get from location L2 to location L3. Figure 4 shows the steps of each plan. The current task is to get from location L4 to location L2. Using the geographic closeness of starting and ending locations alone to judge sim-

Before execution:

- Turn south
- Move south to south side of Birch
- Turn west
- Move west to L2

After execution:

- Turn west
- Move west to L2

Figure 5: Plan C before and after execution

ilarity, plan A appears to be the closest because it shares the same ending location (ROBBIE's retrieval criteria does not include knowledge about reversals of known routes, so plan B does not look similar at all). Plan A is selected and adapted to create plan C (dashed line in Figure 3 and in Figure 5). During this process, the introspective reasoner monitors the system's behavior but detects nothing wrong. When the plan is executed, however, the wasted plan steps will be eliminated: the goal of the first two steps in plan C is to be on the south side of Birch, which is already true, so the steps will be skipped (see Figure 5). When the resulting plan is stored into memory, an introspective failure is detected: an assertion in the model is that the final solution stored will have plan steps which are more similar to the retrieved case than to any other case in memory. In comparing the final plan C to cases A and B in memory, it is clear that plan B has the more similar solution.

In explaining the cause of this assertion failure, ROBBIE reconsiders related assertions assertions in the model, moving up in the hierarchy of assertions to the general assertions that "retrieval will operate successfully." It will consider high-level assertions prior to the general one, such as "the index will select the closest case." That high-level assertion belongs to the Anticipator component; ROBBIE will also move downward from high-level to more specific assertions, including "the index will include all the relevant features to retrieve the closest case." In re-evaluating the last assertion in the context of the failure the system discovers a feature of the cases it had not used before: that each involves moving straight along an east/west street. This shows that the assertion "the index will include all the relevant features to retrieve the closest case" failed. The assertion suggests a repair: add "moves straight on east/west street" to the features used in indexing cases, and re-index memory to include the new feature.

In the future, any problem which involves moving straight along an east/west street will be indexed by the new feature, and will match most closely other cases which also include that feature in their index. Once the introspective reasoner has evaluated and repaired the problem, processing continues normally. Notice that the failure in question here is not a catastrophic one, but it does represent wasted effort on the part of the planner, effort that would otherwise be repeated and compounded in the future.

The situation above is an example of ROBBIE's introspective learning for a single goal. The ramifications of learning a new feature will only become clear over a sequence of goals. In order to study the improvement introspective reasoning provides for ROBBIE, we ran a set of experiments which presented ROBBIE with twenty-six sequences of goals, executing each sequence with and without introspective reasoning. One sequence was carefully designed to be easy for ROBBIE to handle, other sequences were randomly perturbed versions of the first. We measured the number of problems ROBBIE successfully handled for each sequence, and found that in almost every case ROBBIE could handle more problems with introspective learning than without (in one anomalous case the overall performance was so poor that introspective learning could provide no benefit at all). We also measured the percentage of cases in memory which were considered during the the retrieval process, over the sequence of retrievals made in solving the sequence of goals. The percentage considered when introspective reasoning was used dropped significantly below the percentage considered without introspective reasoning. ROBBIE, using introspective reasoning to re-index its memory, considered fewer irrelevant cases *at the same time* as it improved its overall success rate.

Ramifications to general issues

We have now described the ROBBIE system in some detail; we must come back to the issues alluded to briefly above. We will discuss the tradeoff between the generality, and hence transferability, of a self-model framework and the specificity of details the model needs to accurately detect and repair failures. We will also discuss means for evaluating the benefit of learning about reasoning methods. We will describe our attempts to address these issues with the ROBBIE system, sketch our conclusions, and describe how ROBBIE relates to other work in this area.

Generality vs. Specificity

Ideally one could develop a framework for reasoning about mental processes that could be transferred with minor changes to provide self-models for a wide array of underlying systems and tasks (vision, planning, etc.) and for a wide variety of modeling tasks (modeling others' reasoning, explaining reasoning behavior, analyzing its own actions, etc.). While we must admit that such a universal framework is, at least now, out of reach, it is certainly possible to share higher-level insights about mental reasoning, and to develop specific frameworks for more limited tasks and domains. There will be commonalities among the kinds of knowledge, and the useful forms for representing that knowledge, needed to reason about mental actions. Beyond that, it seems reasonable to expect more concrete sharing of model forms and knowledge within a particular kind of self-modeling task.

In developing models of introspection, we will be torn between our desires for transferable models and the reality that a model must include a great deal of system-specific knowledge. Developing approaches that maintain the generality of model as much as possible means focusing on separating details from the functioning of the model, keeping mechanisms and vocabulary used as independent as possible and emphasizing the kind of knowledge needed. Specifying classes of knowledge and useful organizations of that knowledge for describing mental actions will provide the largest gain across modeling tasks.

The problem of integrating a general approach to self-modeling with the details needed to use the model has been one we have tried to address with ROBBIE from the beginning. We designed general mechanisms for monitoring the underlying reasoning and accessing the declarative model which depend in no way on the contents of that model. We are developing a general vocabulary for describing the assertions in the model to complete the generality of the mechanisms. Within this framework a model may be constructed for a very different system sharing little in common with the implemented one. Keeping a hierarchy of assertions, and organizing them by component, allows substitution for pieces of the model for a new system without requiring a completely new model. For example, a CBR system could keep the upper tiers of the model for each component similar to ROBBIE's, adding only new lower-level details, or a variation on ROBBIE which used a different adaptation mechanism could substitute new assertions for that component alone.

Of perhaps greater importance in terms of transferability is what we now understand about the kinds of knowledge and the model structure required for this task. In developing a model for this system we also develop a template for what to include in models of other systems; ROBBIE's model demonstrates the value of incorporating multiple levels of knowledge about reasoning tasks. The ROBBIE system's diagnosis capabilities were improved by having high-level knowledge that provided a general flow of control and information, along with specific details about the system's operation (tied into that higher level). Considering high-level assertions when assigning blame leads the system to consideration of other assertions distant in terms of the reasoning trace but close in terms of the flow of control. The system should more easily trace the reasoning behavior from a detected failure back to the original cause. In a similar way, distinguishing different kinds of relationships between pieces of knowledge focuses the model on the most relevant pieces; in ROBBIE, the model includes specification and abstraction links, links that indicate the sequence of reasoning, and causal links that connect assertions likely to fail or succeed together. The model could choose to follow specification links when trying to determine a repair, or could avoid testing assertions which are specifications of a high-level assertion that has *not* failed. A model without distinct connections between assertions could not as accurately gauge which assertions are relevant under a given set of circumstances.

Many other systems have also approached the problem of generality of mechanism and transferability. Cox & Freed (1994) identify knowledge about how general and specific knowledge combines as a key element for a self-reasoning system. Freed's RAPTER (Freed & Collins, 1994b) uses a general set of representations for expectations and repairs, and a general mechanism to manipulate them, while the content of its representations is specific to the RAPTER system. Stroulia's Autognostic (Stroulia & Goel, 1994) applies an existing kind of model (used for modeling physical machines) to implement a self-model and successfully applied the model and mechanisms to two independent systems (Kritik2 (Stroulia & Goel, 1992) and Router (Goel, Callantine, Shankar, & Chandrasekaran, 1991)). Meta-AQUA (Ram & Cox, 1994) uses abstract descriptions of reasoning traces that might arise under any similar reasoning/explanation task.

Evaluating self-modeling systems

It is often problematic in AI to explain exactly what a given system has accomplished besides showing *some* implementation is possible. It is important to demonstrate the advantages of any learning system in terms of the breadth of problems it can solve and the applicability of its ideas in general. At this point, attempts to evaluate introspective reasoning have been limited; we have, however, made an effort to evaluate ROBBIE's mechanisms and performance.

We must determine when using a self-model provides a benefit, and how to demonstrate the extent of that benefit. That benefit may be in advancing our knowledge of what self-reasoning entails and the ramifications for mental modeling in general. The benefit may also lie on the practical side as well: systems with the power to improve their own mechanisms should solve more problems, solve problems more effectively, produce better solutions, and respond more flexibly to novel situations than their non-introspective counterparts. The expense of modeling reasoning behavior makes evaluating its success as a practical tool of particular importance.

How to measure the performance of an introspective learning system is itself a difficult question and may depend on the system; possible measures include: the breadth and number of problems solved that were impossible previously, the speed and efficiency of the reasoning process and the solutions produced, and many others. Many systems which use a model of reasoning, including ROBBIE, are two-level systems which make a relatively firm distinction between the reasoning being modeled and the reasoning used to do the modeling; one possible evaluation method is to compare the performance of the bottom-level system with the system as a whole. As a qualitative evaluation we can ask if a system like ROBBIE *detects* the "right" failures, *assigns* blame correctly, and *repairs* the system the "right" way. Other work has been less explicit about concrete means of evaluating systems. Cox (1995) has described classes of reasoning behavior and failures that people experience, and that systems which model reasoning behavior should address; that set provides a qualitative guide for judging models of reasoning. Autognostic (Stroulia & Goel, 1994) provides another kind of evaluation by directly proving the applicability of its model to different underlying systems.

We have begun evaluating ROBBIE using a practically-oriented criterion: the addition of introspective reasoning should produce quantitative as well as qualitative improvements in the performance of the overall system. We are in the process of performing extensive experiments to test ROBBIE's performance over long sequences of problems. By collecting statistics on the success of the system with and without introspective learning, we can quantify its effect. Some tentative and preliminary results are in (Fox & Leake, 1994). We have completed one set of experiments (described above) which used the number of successful cases over a sequence and the percentage of cases in memory considered during retrieval to reveal differences in ROBBIE performance with and without introspective reasoning. Initial results of that experiment are encouraging.

Focusing too heavily on quantitative measures may overlook some important features of introspective reasoning; it is difficult to quantify the quality of a solution, or the elegance

of the reasoning that created it. We must be aware of and seek out those more qualitative benefits as well. We may find objective measures of solution quality through common sense in some domains, through comparisons with human-created solutions, or through surveys eliciting quality judgements. Elegance of reasoning is an even more subjective issue, but by similar methods some objective judgement can be reached.

Conclusions

The ROBBIE system, while still incomplete, addresses several important issues for modeling reasoning behavior, and introspective learning in particular. Our conclusions about the structural requirements of ROBBIE's model should be applicable to a general model of reasoning, and the approach to preserving the re-usability of the model may also provide pointers for future work on the transfer of reasoning knowledge. ROBBIE benefited from using multiple levels of knowledge to focus on the most relevant portions of the model; determining what the important levels of knowledge are and how multiple levels affect reasoning models may be beneficial to a wide range of models of reasoning behavior.

In developing a model of the reasoning process, we must strike a balance between the generality and transferability of the model and the specific knowledge required to detect and specify repairs. The ROBBIE system uses a hierarchical model accessed by system independent mechanisms in order to find that balance. To achieve generality, mechanisms for introspective reasoning should work with any set of assertions needed for a system, requiring a general vocabulary or framework for a vocabulary. Separating assertions which make statements about the general kind of underlying system (here, case-based reasoning) from those that refer to implementation or knowledge details of the specific system makes it easier to convert the model to apply to a new but similar system. We claim that a model *must* include knowledge about the reasoning process at multiple levels of abstraction; a high-level description tied to lower-level details. Doing so helps us to keep the generality of the model and also allows us to use the model at the "right" level for diagnosis by using abstract descriptions to trace the general flow of control and knowledge rather than plodding through every detailed step of the reasoning process. Using the right level of abstraction may focus the diagnosis on promising areas of the model while avoiding unnecessary or unpromising details.

We claim that the problem of evaluating models of reasoning behavior must be addressed because of the potential expense of such models. We can choose to evaluate a system in terms of its benefit as a model of mental actions: what we learn about possible model structures and knowledge needs provide one kind of justification, or the extent to which a model covers the scope of introspective reasoning for the task. We may also evaluate the practical benefits of using a model of reasoning behavior. For the purposes of a system using the model for self-repair, we may judge the quality of the overall system compared to one without learning, or use a qualitative gauge of the repairs made to the system. To judge the quality of the overall system, various measures might be proposed: breadth of problems solved, quality of solutions, speed of processing, and so forth. We have begun this process by trying to find some quantitative measures of ROBBIE's improvement when introspective learning is enabled, while also considering ways to express qualitative measures such as "improving in the right way" or "learning from the right failures."

An issue at the heart of self-modeling systems is the question of what kinds of knowledge are required for the system to perform its tasks, and how that knowledge is to be represented. While this is an ongoing research issue, we have proposed a structure for self-modeling that allows for flexibility of application and is designed to allow for transfer of some part of the model to new applications.

Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. IRI-9409348.

References

Alterman, R. (1986). An adaptive planner. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pp. 65–69 Philadelphia, PA. AAAI.

Birnbaum, L., Collins, G., Brand, M., Freed, M., Krulwich, B., & Pryor, L. (1991). A model-based approach to the construction of adaptive case-based planning systems. In Bareiss, R. (Ed.), *Proceedings of the Case-Based Reasoning Workshop*, pp. 215–224 San Mateo. DARPA, Morgan Kaufmann, Inc.

Birnbaum, L., Collins, G., Freed, M., & Krulwich, B. (1990). Model-based diagnosis of planning failures. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pp. 318–323 Boston, MA. AAAI.

Collins, G., Birnbaum, L., Krulwich, B., & Freed, M. (1993). The role of self-models in learning to plan. In *Foundations of Knowledge Aquisition: Machine Learning*, pp. 83–116. Kluwer Academic Publishers.

Cox, M. (1995). Representing mental events (or the lack thereof). In *Proceedings of the 1995 AAAI Spring Symposium on Representing Mental States and Mechanisms*. (in press).

Cox, M. & Freed, M. (1994). Using knowledge of cognitive behavior to learn from failure. In *Proceedings of the Seventh International Conference on Systems Research, Informatics and Cybernetics*, pp. 142–147 Baden-Baden, Germany.

Firby, R. J. (1989). *Adaptive Execution in Complex Dynamic Worlds*. Ph.D. thesis, Yale University, Computer Science Department. Technical Report 672.

Fox, S. & Leake, D. (1994). Using introspective reasoning to guide index refinement in case-based reasoning. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, pp. 324–329 Atlanta, GA. Lawrence Erlbaum Associates.

Freed, M. & Collins, G. (1994a). Adapting routines to improve task coordination. In *Proceedings of the 1994 Conference on AI Planning Systems*, pp. 255–259.

Freed, M. & Collins, G. (1994b). Learning to prevent task interactions. In desJardins, M. & Ram, A. (Eds.), *Proceedings of the 1994 AAAI Spring Symposium on Goal-driven Learning*, pp. 28–35. AAAI Press.

Goel, A., Callantine, T., Shankar, M., & Chandrasekaran, B. (1991). Representation, organization, and use of topographic models of physical spaces for route planning. In *Proceedings of the Seventh IEEE Conference on AI Applications*, pp. 308–314. IEEE Computer Society Press.

Hammond, C. (1989). *Case-Based Planning: Viewing Planning as a Memory Task*. Academic Press, San Diego.

Kolodner, J. (1993). *Case-Based Reasoning*. Morgan Kaufman, San Mateo, CA.

Krulwich, B., Birnbaum, L., & Collins, G. (1992). Learning several lessons from one experience. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pp. 242–247 Bloomington, IN. Cognitive Science Society.

Leake, D. (1992). *Evaluating Explanations: A Content Theory*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Oehlmann, R., Edwards, P., & Sleeman, D. (1994). Changing the viewpoint: re-indexing by introspective questioning. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, pp. 675–680. Lawrence Erlbaum Associates.

Oehlmann, R., Edwards, P., & Sleeman, D. (1995). Introspection planning: representing metacognitive experience. In *Proceedings of the 1995 AAAI Spring Symposium on Representing Mental States and Mechanisms*. (in press).

Ram, A. (1989). *Question-driven understanding: An integrated theory of story understanding, memory and learning*. Ph.D. thesis, Yale University, New Haven, CT. Computer Science Department Technical Report 710.

Ram, A. & Cox, M. (1994). Introspective reasoning using meta-explanations for multistrategy learning. In Michalski, R. & Tecuci, G. (Eds.), *Machine Learning: A multistrategy approach Vol. IV*, pp. 349–377. Morgan Kaufmann.

Riesbeck, C. (1981). Failure-driven reminding for incremental learning. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pp. 115–120 Vancouver, B.C. IJCAI.

Schank, R. (1986). *Explanation Patterns: Understanding Mechanically and Creatively*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Stroulia, E. & Goel, A. (1992). Generic teleological mechanisms and their use in case adaptation. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pp. 319–324 Bloomington, IN. Cognitive Science Society.

Stroulia, E. & Goel, A. (1994). Task structures: what to learn?. In desJardins, M. & Ram, A. (Eds.), *Proceedings of the 1994 AAAI Spring Symposium on Goal-driven Learning*, pp. 112–121. AAAI Press.