11 Using Introspective Reasoning to Improve CBR System Performance

Josep Lluís Arcos, Oğuz Mülâyim, David B. Leake

When AI technologies are applied to real-world problems, it is often difficult for developers to anticipate all possible eventualities. Especially in long-lived systems, changing circumstances may require changes not only to domain knowledge but also to the reasoning process that brings it to bear. This requires *introspective reasoning*, metareasoning by a system about its own internal reasoning processes. This chapter investigates applying introspective reasoning to improve the performance of a case-based reasoning system, by guiding learning to improve how a case-based reasoning system applies its cases.

Case-based reasoning (CBR) is a problem-solving methodology that exploits prior experiences when solving new problems, retrieving relevantly similar cases and adapting them to fit new needs (for an overview and survey, see Mantaras et al., 2005). Many CBR systems store each newly solved problem and its solution as a new case for future use, enabling them to continuously improve their case knowledge. Nevertheless, the success of a CBR system depends not only on its cases, but also on its ability to use those cases appropriately in new situations, which depends on factors such as the system's similarity measure and the case adaptation mechanism. Consequently, it is desirable to enable CBR systems to improve the knowledge and processes by which they bring their cases to bear.

Metareasoning techniques provide a promising basis for self-improving systems (see Anderson & Oates, 2007, or Cox, 2005, for recent reviews). As described by Cox and Raja (this vol., chap. 1), the metareasoning approach incorporates a metareasoning layer, with monitoring and control capabilities over the reasoning process, to adjust that reasoning process as needed. Introspective learning techniques have used *self-models* as a way to determine when, what, and how to improve the reasoning of systems. Here we focus on how a self-model may be exploited; an open challenge is how to provide capabilities for extending and refining self-models that are themselves imperfect or incomplete (Leake & Wilson, 2008).

Previous research on introspective CBR has shown that metareasoning can enable a CBR system to learn by refining its own reasoning process. That work has tended to

apply the introspective approach only to a single aspect of the CBR system, for example, to adjust the indices used for retrieval. This chapter presents research on developing an introspective reasoning model enabling CBR systems to autonomously learn to improve multiple facets of their reasoning processes.

The remainder of this chapter describes an approach in which an introspective reasoner monitors the CBR process with the goal of adjusting the retrieval and reuse strategies of the system to improve solution quality. Novel aspects of this approach, compared to previous work on introspective reasoning for CBR, include that it applies a unified model for improving the two main stages of the CBR process, that a single failure may prompt multiple forms of learning, and that it performs internal tests to empirically assess the value of changes proposed by the introspective reasoner, to determine which ones should be retained.

The next section discusses previous work on introspective learning for case-based reasoning. The following section presents a detailed description of our approach and its implementation. The approach has been evaluated on problems from a fielded industrial application for design of pollution control equipment, for which we provide results in the next section. Before concluding the chapter, we put our model into context with respect to the metareasoning models discussed in chapter 1 of this volume. In the last section we present our conclusions and directions for future research.

Related Work

Birnbaum et al. (1991) first proposed the use of self-models within case-based reasoning. Work by Cox and Ram (1999) develops a set of general approaches to introspective reasoning and learning that automatically select the appropriate learning algorithms when reasoning failures arise. Their work defines a taxonomy of causes of reasoning failures and proposes a taxonomy of learning goals, used for analyzing the traces of reasoning failures and responding to them. Here case-based reasoning is a vehicle for supporting introspective reasoning: CBR is used to explain reasoning failures and generate learning goals.

A number of studies apply introspective approaches to improve the performance of CBR systems. Leake (1996) identifies the knowledge sources a CBR system uses in its reasoning process and the required self-knowledge about these sources, and provides examples of refinement of retrieval knowledge using model-based reasoning and of acquisition of adaptation knowledge by search plans. Fox and Leake (2001) developed a system inspired by the Birnbaum et al. proposal to refine index selection for case-based reasoners. Fox and Leake's work develops a declarative model for describing the expectations for correct reasoning behavior, and applies that model to detect and diagnose reasoning failures. When the introspective reasoner is able to identify the

feature that caused the failure, the system's memory is reindexed, resulting in significant performance improvement. The DIAL system (Leake et al., 1995) improves case adaptation using introspection. This research focuses on improving the performance of the system by storing the traces of successful adaptation transformations and memory search paths for future reuse. Likewise, Craw (2006) proposes an introspective learning approach for acquiring adaptation knowledge, making it closely related to our work. However, a key difference is that their learning step uses the accumulated case base as training data for adaptation learning, in contrast to our approach of incrementally refining adaptation knowledge in response to failures for individual problems.

Arcos (2004) presents a CBR approach for improving solution quality in evolving environments. His work focuses on improving the quality of solutions for problems that arise only occasionally, by analyzing how the solutions of more typical problems change over time. Arcos's algorithm improves the performance of the system by exploiting the neighborhoods in the solution space but, unlike the model presented in this chapter, learns only from success.

The REM reasoning shell (Murdock & Goel, 2008) presents a meta-case-based reasoning technique for self-adaptation. The goal of REM is the design of agents able to solve new tasks by adapting their own reasoning processes. Meta-case-based reasoning is used for generating new task-method decomposition plans. Because the goal in REM is the assembly of CBR reasoning components, the metamodel is focused on describing the components in terms of their requirements and their effects. In contrast, our model is focused on describing the expected correct properties of the components and their possible reasoning failures.

Introspective reasoning to repair problems may also be seen as related to the use of confidence measures for assessing the quality of the solutions proposed by a CBR system (Cheetham & Price, 2004; Delany et al., 2005). Confidence measures provide expectations about the appropriateness of proposed solutions. A high confidence solution that is determined to be erroneous reveals a failure of the reasoning process used to form the prediction, which points to the need to refine the self-model. The unexpected success in a low-confidence solution may do so as well. Nevertheless, because confidence measures provide no explanations for their assessments, they are not helpful for revealing the origin of the reasoning failure, making their failures hard to use to guide repairs.

Introspective Reasoning Approach

The goal of our introspective reasoning system is to detect reasoning failures and to refine the function of reasoning mechanisms, to improve system performance on future problems. To achieve this goal, the introspective reasoner monitors the

reasoning process, determines the possible causes of its failures, and performs actions that will affect future reasoning processes.

To give our system criteria for evaluating its case-based reasoning performance, we have created a model of the correctly-functioning CBR process itself, together with a taxonomy of reasoning failures. Failures of a CBR system's reasoning process are modeled as conflicts between observed system performance and predictions from the model. These failures, in turn, are related to possible learning goals. Achieving these goals repairs the underlying cause of the failure.

As illustrated in the bottom portion of figure 11.1, the case-based reasoning process consists of four steps: (1) *case retrieval/similarity assessment*, which determines which cases address problems most similar to the current problem, to identify them as starting points for solving the new problem; (2) *case adaptation*, which forms a new solution by adapting/combining solutions of the retrieved problems; (3) *case revision*, which evaluates and adjusts the adapted solution; and (4) *case retention*, in which the system learns from the situation by storing the result as a new case for future use.

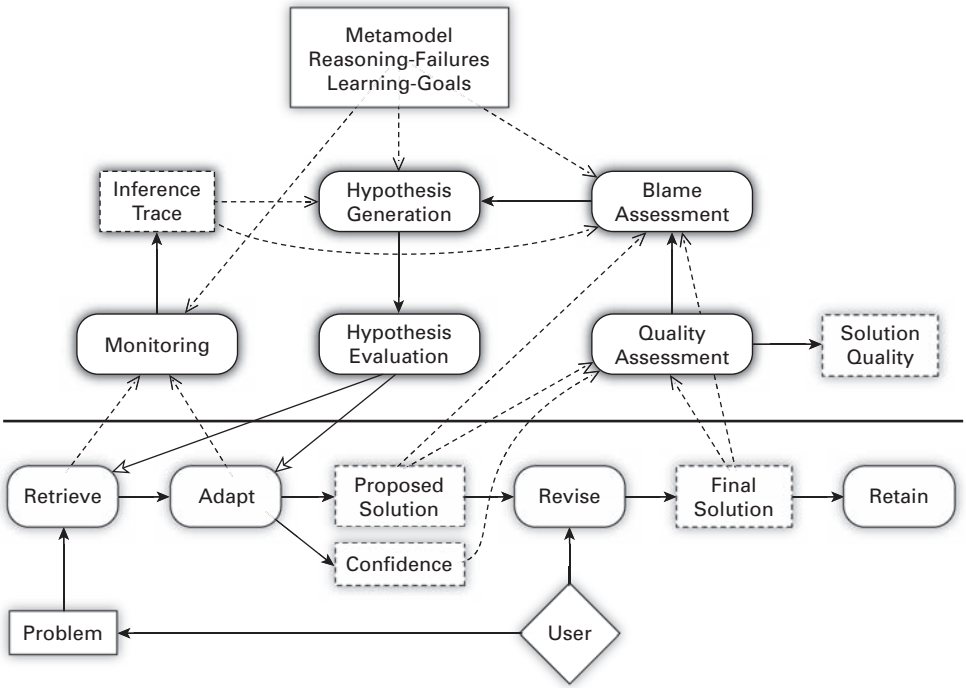

Figure 11.1
Introspective reasoner components. The horizontal line divides the CBR process (bottom) and the introspective reasoner (top).

Reasoning failures may be revealed by either of two types of situation: (i) when the retrieval or adaptation step is unable to propose a solution, or (ii) when the solution proposed by the system differs from the desired final solution. Failures of the retrieval or adaptation steps are identified directly by contrasting their performance with model predictions. The second type of failure can be detected by monitoring the revision step. In CBR systems, the revision step often involves interaction with the user to determine the final solution. This interaction provides a feedback mechanism for assessing the real quality of the solution initially proposed.

For each of the four CBR steps, the model encodes expectations, and the expectations are associated with learning goals that are triggered if the expectations are violated. For example, the expected behavior of the similarity assessment step is to rank the retrieved cases correctly. If they are ranked incorrectly, the failure may be due to using an inappropriate weighting when similarity assessments along different dimensions are aggregated. Consequently, a possible strategy for solving the failure is to refine the weight model, and a corresponding learning goal is to learn new weightings.

Our model is domain independent, that is, it is focused on the general case-based reasoning process for retrieval and adaptation, rather than on specific details of those processes for any particular domain. The model deals with three types of knowledge: indexing knowledge, ranking knowledge, and adaptation knowledge. To apply the model to any concrete application, domain-specific retrieval and adaptation mechanisms must be linked to the model.

Indexing knowledge determines the subspace of the case base considered relevant to a given problem. Ranking knowledge identifies the features considered most relevant to determining similarity, given a collection of retrieved cases. Adaptation knowledge defines transformative and/or generative operations for fitting previous solutions to a current problem.

Our approach is shaped by two assumptions about the underlying CBR system. The first is that the system is initially provided with general retrieval and adaptation mechanisms, which apply uniform criteria to problems throughout the problem space. This is a common property of many case-based reasoning systems, but experience developing CBR systems has shown that this uniform processing may result in suboptimal processing, in turn resulting in the generation of low-quality solutions. Consequently, one of the focuses of our approach is to address this problem: One of the learning goals of the introspective reasoner is to determine the real scope of cases, to weight the different ranking criteria, and to refine the adaptation model for different problem space regions.

The taxonomy defined for the learning goals borrows partially from the taxonomy of learning goals proposed by Cox and Ram (1999). Nevertheless, in our approach the learning goals are oriented specifically toward refining the CBR process. For example,

determining the scope of cases is modeled in terms of differentiation/reconciliation goals, whereas improving the ranking criteria is modeled in terms of refinement/organization goals.

The second working assumption is that the CBR system is able to determine internal confidence estimates for its solutions to new problems. Because confidence assessment will be domain specific, it is not part of our general model. In the application we consider, the system always serves in an advisory role to an engineer, who assesses the system-generated solution before applying it. The engineer's assessment provides a natural source of feedback for judging whether the system's confidence value was appropriate.

Rather than reasoning about numeric confidence values, we deal with confidence using three qualitative values: *low confidence*, *medium confidence*, and *high confidence*. The mapping to the numeric intervals that represent the qualitative values must be defined in each application. For instance, in our chemical application, due to the importance of safety constraints in the chemical processes, high confidence is ascribed to values greater than 0.8 on a 0–1 scale, and the threshold for low confidence is 0.6.

The system's introspective reasoning is organized into five tasks: (1) the *monitoring* task, in charge of maintaining a trace of the CBR process; (2) the *quality assessment* task, which analyzes the quality of the solutions proposed by the system; (3) the *blame assessment* task, responsible for identifying the reasoning failures; (4) the *hypothesis generation* task, in charge of proposing learning goals; and (5) the *hypothesis evaluation* task, which assesses the impact of proposed improvements on solution generation.

Figure 11.1 depicts the introspective reasoning components. The horizontal line divides the CBR process (bottom) from the Introspective Reasoner (top). Rounded boxes represent inference processes; dashed boxes represent knowledge generated by inference; dashed lines show knowledge dependencies; black-tipped arrows show inference flows; and hollow-tipped arrows denote control relationships.

Monitoring

The monitoring task tracks the case-based reasoning process. For each problem solved by the CBR system, the monitor generates a trace containing: (1) the cases retrieved, with a link to the indexing knowledge responsible for the retrieval; (2) the ranking criteria applied to the cases, together with the values that each criterion produced and the final ranking; and (3) the adaptation operators that were applied, with the sources to which they were applied (the cases used) and the target changes produced (the solution features).

Note that our model does not require that the adaptation step use only a single case, nor that all the retrieved cases be involved in all adaptations; any such constraints depend on specific applications, independent of the general model. Similarly,

our model distinguishes application of indexing criteria and ranking criteria as two subprocesses involved in the retrieval step, but it does not require that they be decoupled in the implementation being monitored. For instance, a K-nearest neighbor approach (Cover & Hart, 1967) uses the value of K to determine the number of cases considered and uses the distance measure as a ranking criterion. Other approaches might separate indexing and ranking by, for example, using crude criteria for indexing and finer-grained criteria for case ranking.

Quality Assessment

When the user's final solution is provided to the system, quality assessment is triggered to determine the "real" quality of the system-generated solution, by analyzing the differences between the system's proposed solution and the final solution. Quality assessment provides a result in qualitative terms: *low quality*, *medium quality*, or *high quality*.

Given the system's initial confidence assessment and the final quality assessment, the introspective reasoner fires learning mechanisms when there is a mismatch between the two. There are two main types of possible mismatches. When the confidence was high but the actual quality is low, the conflict points to a failure at the retrieval stage, because the confidence of a solution has a strong relationship with the coverage of the retrieved cases (Cheetham, 2000).

On the other hand, when the confidence was low but the quality is demonstrated to be high, the unexpected success may be due either to low coverage from cases (none of the system's cases appeared highly relevant) or to bad ranking of the retrieved cases (the most relevant cases were not considered, due to a failure of the ranking polices to identify them). When the mismatch between the confidence and the quality assessments is small (i.e., high versus medium, medium versus high, medium versus low, and low versus medium) it may suggest a failure in the adaptation stage.

Blame Assessment

Blame assessment starts by identifying the source of the failure. It takes as input the differences between the solution and expected result, and tries to relate the solution differences to the retrieval or the adaptation mechanisms. The system searches the taxonomy of reasoning failures and selects those that apply to the observed solution differences. For instance, when a final solution is radically different from the solution proposed by the system, the failure may be caused by the indexing knowledge, that is, either the relevant precedents have not been retrieved or too many cases have been retrieved.

Search for applicable failures in the failure taxonomy uses the trace generated by the monitoring module. It starts by analyzing the index failures. There are three types

of index failures: *wrong index*, *broad index*, and *narrow index*. When none of the retrieved cases has a solution close to the current solution, the wrong index failure is selected. A broad index failure is selected when many cases are retrieved and their solutions are diverse. On the other hand, when a small set of cases is retrieved, the narrow index failure is selected.

Ranking failures are identified by comparing the retrieval rankings with the solution differences they generate. Examples of ranking failures are *inappropriate ranking scheme*, *overestimated weights*, and *underestimated weights*.

Adaptation failures are identified by linking the solution differences to the adaptation operators stored in the monitoring trace. When adaptation uses interpolation, adaptation failures originate in inappropriate interpolation policies.

Because the introspective reasoner will often not be able to determine a unique source for a failure, all the possible causally supported failures are chosen, resulting in multiple types of learning goals from a single failure.

Hypothesis Generation

The fourth reasoning stage, hypothesis generation, identifies the learning goals related to the reasoning failures selected in the blame assignment stage. Each failure may be associated with more than one learning goal. For instance, there are multiple ways of solving overestimated weights. For each learning goal, a set of plausible local retrieval/adaptation changes in the active policies is generated, using a predefined taxonomy.

Table 11.1 shows some of the types of hypotheses generated to explain failures in the retrieval and adaptation stages. The changes must be local because their applicability is constrained to the neighborhood of the current problem. For instance, when a refinement goal is selected for the adaptation knowledge, an adaptation adjustment is selected from a predefined collection of tuning actions depending on the nature of the original adaptation. Specifically, when adaptations are related to numerical features the tuning actions determine different types of numerical interpolations. The two main types of changes in numeric features affect the *shape* and *slope* of the interpolation curve.

Table 11.1
Examples of types of hypotheses used by the introspective reasoner

Failure	Learning Goal
Missing index	Create index
Broad index	Refine index
Underestimated weight	Adjust weighting
Inappropriate interpolation	Change shape Increase slope

Hypothesis Evaluation

The fifth reasoning stage, hypothesis evaluation, evaluates the impact of introducing retrieval/adaptation changes. Because the introspective reasoner does not have a complete model of the inference process, it is not possible for it to definitively predict the effects of changes. Consequently, before altering the CBR system, some empirical evidence about the impact of the change must be obtained. In our current design this evidence is obtained by re-solving the problem, applying each proposed change and evaluating its impact. Retrieval/adaptation changes that improve the quality of the solution are incorporated into the CBR inference mechanisms. Note that when the introspective reasoner provides a problem to the CBR system for testing purposes, the case retention step is deactivated.

Experiments

We have tested the introspective reasoner as an extension to a fielded industrial design application. We have developed a case-based reasoning system for aiding engineers in the design of gas treatment plants for the control of atmospheric pollution due to corrosive residual gases that contain vapors, mists, and dusts of industrial origin (Arcos, 2001). A central difficulty for designing gas treatment plants is the lack of a complete model of the chemical reactions involved in the treatment processes. Consequently, the expertise acquired by engineers from their practical experience is essential for solving new problems. Engineers have many preferences and deep chemical knowledge, but our interactions have shown that it is hard for them to determine in advance (i.e., without a new specific problem at hand) the scope and applicability of previous cases. They apply some general criteria concerning factors such as cost and safety conditions, but other criteria depend on specific working conditions of the treatment process.

On the other hand, because engineers make daily use of the application system to provide the final solutions to customers, the system has the opportunity to compare its proposed solutions with the solutions finally delivered. Thus, we have the opportunity to assess the impact of the introspective reasoner on the quality of the solutions proposed by the CBR system.

Applying the CBR Process

The inference process in this design application is decomposed into three main stages: (1) selecting the class of chemical process to be realized; (2) selecting the major equipment to be used; and (3) determining the values for the parameters for each piece of equipment.

The quality of proposed solutions is computed automatically, by comparing the proposed solution to the solution applied by the experts at these three different stages.

Mismatches at earlier steps are more serious than at later ones. For example, except in the case of underspecified problems, a mismatch with the class of the chemical process would indicate a very low-quality solution.

The retrieval and adaptation steps were designed taking into account the three knowledge sources described in the previous section: indexing criteria, ranking criteria, and adaptation operators. Here the problem features are related to the detected pollutants, the industrial origin of the pollutants, and working conditions for the pollution-control equipment (flow, concentrations, temperature). Indexing criteria determine the conditions for retrieving cases. The main indexing criteria are related to the initially defined chemical relations among pollutants. Ranking criteria determine a preference model defined as a partial order. Initially, the preferences are homogeneous for the whole problem space. Throughout the experiments, the introspective reasoner automatically refines the initial model.

Reasoning failures originate from situations in which the criteria do not properly identify the main pollutants or critical working conditions. The consequences are manifested in solutions for which the proposed chemical process is not correct or there are inappropriate washing liquids, or by mismatches on equipment parameters.

Testing Scenario

The design application can solve a broad range of problems. However, to test the effects of introspective reasoning for learning to handle novel situations, it is desirable to focus the evaluation on sets of frequently occurring problems which share at least a pollutant (the minimal indexing criterion), in order to have reuse. On the other hand, it is necessary to have sufficient diversity: Good performance on quasi-identical problems can be obtained by case learning alone, so such problems do not generate opportunities for the introspective reasoner.

We decided to focus the evaluation of the system on problems with the presence of hydrogen sulfide, a toxic gas produced by industrial processes such as wastewater treatment. From the existing application, we had access to the 510 such solved problems, ordered chronologically. We divided the problems into two sets: 300 initial system cases and 210 testing problems.

To evaluate the contribution of the introspective reasoner we performed an ablation study, comparing the performance of the system when presenting the problems sequentially for five different reasoning strategies. In addition to testing inputs in chronological order, we repeated the experiments ten times with random orders for the testing problems, to assess the sensitivity of learning to problem ordering. The tested reasoning strategies are the following:

No-Retain, a strategy that solved the problems without introspective reasoning and without incorporating the solved cases into the case memory;

Retain, which solved the problems without introspective reasoning but incorporated solved cases into the system (the only learning normally done by CBR systems);
Int-Retr, which combined *Retain* with introspective reasoning only for retrieval refinement;
Int-Adapt, which combined *Retain* with introspective reasoning only for adaptation refinement; and
Int-Compl, which combined Retain with introspective reasoning for both retrieval refinement and adaptation refinement.

Results

Table 11.2 shows the results of the evaluation for chronological problem presentation (results for random ordering were similar). Results support that the storage of solved problems—case learning alone—improves the performance of the system, but also show that this policy is not sufficient. Although the number of high-confidence solutions increased significantly, the decrease of low-quality solutions is not statistically significant (see second column in table 11.2).

A second conclusion from the results is that the main contribution of using introspection to refine retrieval knowledge is to reduce the number of low-quality solutions (a 36.67 percent reduction compared with case learning alone). In our design application this improvement is achieved by providing more accurate ranking policies for determining the chemical process to be realized.

The main contribution of using introspection for refining adaptation knowledge (see fourth column in table 11.2) is an increase in the number of high-quality solutions (a 12.5 percent increment from *Retain*). In our task, learning more appropriate adaptation policies enables better determination of the different equipment parameters.

Interestingly, when introspection adjusts both retrieval and adaptation (last column in table 11.2), the improvement in the retrieval step has an indirect effect on the adaptation step, increasing the number of high-quality solutions. An intuitive explanation is that better retrieval also facilitates the adaptation process. Thus, using both introspection strategies, the increase in the number of high-quality solutions, with respect to case learning alone, reaches 15.63 percent.

Table 11.2
Average solution quality for all the strategies

	No-Retain	Retain	Intr-Retr	Int-Adap	Int-Compl
High-Quality	23.81%	30.92%	30.95%	34.29%	35.75%
Medium-Quality	59.52%	54.59%	60.00%	52.86%	56.04%
Low-Quality	16.67%	14.49%	9.05%	12.85%	8.21%

Comparing the number of problems that changed their quality of solution, 12 percent of the total solved problems qualitatively increased their solution quality when introspection was used. Solution qualities varied, but the use of introspection did not decrease the solution quality for any problem. Moreover, the reduction in low-quality solutions is statistically significant ($\rho < 0.05$), even though the increase in high-quality solutions is not statistically significant. Consequently, we conclude that the number of problems whose solution quality was improved by the use of introspection is statistically significant.

Table 11.3 summarizes the introspective reasoner's processing. Results summarize the experiments using both introspection strategies, reflecting learning goals triggered from the detection of 135 non-high-confidence solutions. Most activity was focused on ranking and adaptation failures, because these are the most difficult tasks. Note that not all the generated hypotheses were considered useful by the system (see third and fourth columns): revisions to the reasoning process were performed for 17 percent of the instances for which learning goals were triggered. This result illustrates that the introspective reasoner is dealing with partial understanding of the CBR process and that the introspective learner's hypotheses should be tested before being applied.

It is clear that the incorporation of the introspective reasoner entails some computational overhead. However, it does not interfere with normal system performance: The introspective reasoner is triggered only *after* a problem is solved and is a background process without user intervention. Most of the cost of introspective reasoning arises from hypothesis generation. Table 11.3 shows that the ratio between failures and hypotheses generated is 0.6, because only failures highly explained by the model become hypotheses. Consequently, the number of hypotheses to verify is limited.

A risk of triggering metareasoning in response to individual reasoning failures is the possibility of treating exceptions as regular problems. In the current experiments, such situations did not arise, but in general we assume that the user is responsible for recognizing the exceptions. In addition, only taking action in response to clearly identified failures helps the system to avoid reasoning about exceptions.

Table 11.3
Summary of the number of times learning goals are triggered. Occ stands for failure occurrences, Prop stands for hypotheses generated, and Inc stands for changes incorporated into the CBR process

Failures	Occ.	Prop.	Inc.
Indexing Knowledge	12	5	3
Ranking Knowledge	83	41	8
Adaptation Knowledge	74	56	12

Research on humans has shown that introspection may sometimes have negative consequences. Experiments reported by Wilson and Schooler (1991) showed that, when people are asked to think about the reasons for a given decision, their attempt to form plausible explanations for the specific context of the current decision may result in nonoptimal explanations, negatively affecting future decisions. However, such risks do not apply directly to our approach. First, only the changes incorporated into the CBR process affect future decisions, that is, not the exploration of plausible hypotheses. Second, the goal of the hypothesis evaluation process is to verify the effect of candidate changes on the system. Third, the changes incorporated only have local effects.

Relationship to the Metareasoning Manifesto

Compared to the metareasoning models described in chapter 1 of this volume, our approach is closely related to the use of metalevel control to improve the quality of decisions. Taking the "Duality in reasoning and acting" diagram (fig. 1.2) of chapter 1 as a starting point, our approach incorporates some revisions, as illustrated in figure 11.2 and described in the following points.

First at all, at the ground level, our approach adds the user of the system. The role of the user is twofold: the user (1) presents new problems to the system and (2) provides feedback by revising the solution proposed by the object level. This second role is crucial since it allows the metalevel to estimate the performance of the object level.

In our system, the metalevel continuously monitors the object level (the case-based reasoning process) and assesses the quality of the solutions proposed by the reasoner (using the quality assessment module). The feedback of the user's final solution is exploited to assess the mismatch between system's expectations for its solution (the solution proposed at the object level) and the correct solution (the solution obtained from the ground level).

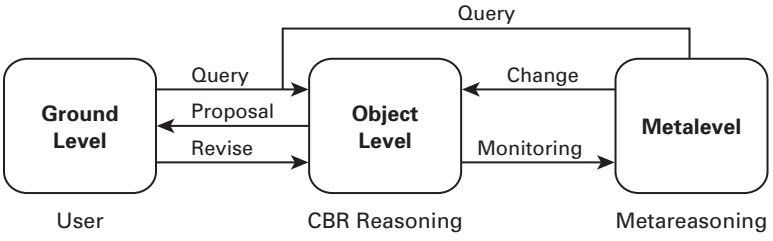

Figure 11.2
Relating our model with existing metareasoning models.

We note the importance of the hypothesis evaluation step. Because the introspective reasoner cannot completely predict the effects of changing the reasoning level, the hypothesis evaluation phase acts as an online trainer. Thus, the metalevel, analogously to the ground level, has the ability to require that the object level solve new problems (top-most query arrow in figure 11.2). Moreover, when the metalevel is testing the performance of the object level, it can temporally deactivate the retention step (in our experiments this is achieved by activating the No-Retain policy).

The control of the object level is achieved by acting over three types of knowledge components used in the reasoning process at the object level: indexing knowledge, ranking knowledge, and adaptation knowledge.

Conclusions

This chapter presented a new introspective model for autonomously improving the performance of a CBR system by reasoning about system problem-solving failures. To achieve this goal, the introspective reasoner monitors the reasoning process, determines the causes of the failures, and performs actions that will affect future reasoning processes.

The introspective level reasons about the reasoning at the object level and about alternative choices to improve the object-level reasoning. Specifically, it relies on a causal model of the correctly functioning retrieval and adaptation stages of CBR. Failures of a CBR system's reasoning process are modeled as conflicts between observed system performance and predictions from the causal model. The sources of these conflicts are identified and associated learning goals are introduced, sometimes triggering multiple types of learning. As a result the case-based reasoning process is improved for future problem solving.

We have tested the introspective reasoner in a fielded industrial design application. Experiments show that the use of the introspective reasoner improved the performance of the system. Introspection-based refinements of retrieval knowledge reduced the number of low-quality solutions; refinements to adaptation knowledge increased high-quality solutions. Moreover, the combination of both is able to generate more high-quality solutions.

Because we tested the introspective prototype in a fielded application previously developed by one of the authors, we had the opportunity to deeply analyze and compare the performance of both systems. The fielded application was developed by introducing many ad hoc mechanisms (concerning similarity and adaptation), whereas the introspective prototype was initially provided with only some broad mechanisms. Over the course of the experiments, the introspective prototype was able to refine its initial reasoning and reach a performance comparable to that of the fielded application. Thus, one lesson of this research is that a domain-independent introspective

reasoner is a powerful tool that facilitates the design of a CBR system by providing a mechanism that can autonomously improve the system's reasoning when required.

Because our model of the CBR reasoning process is domain independent, it can be applied in other domains. The engineering effort for incorporating the metareasoning component to other domains would be concentrated on linking domain-specific aspects of the CBR reasoning process to the appropriate parts in the model (retrieval, adaptation, and revision models). The application of the metareasoning component to other design domains would provide an opportunity to validate the completeness of the taxonomies of reasoning failures and learning goals. Our current work aims at exploring the generality of our approach.

Acknowledgments

This research has been partially supported by the Spanish Ministry of Education and Science project MID-CBR (TIN2006-15140-C03-01), EU-FEDER funds, and by the Generalitat de Catalunya under the grant 2005-SGR-00093. This work has been conducted in the framework of the Doctoral Program in Computer Science of the Universitat Autònoma de Barcelona. This material is also based upon work supported by the National Science Foundation under grant No. OCI-0721674.

References

Anderson, M. L., & Oates, T. (2007). A review of recent research in metareasoning and metalearning. *AI Magazine*, *28*(1), 7–16.

Arcos, J. L. (2001). T-air: A case-based reasoning system for designing chemical absorption plants. In D.W. Aha & I. Watson (Eds.), *Case-based reasoning research and development*. No. 2080 in Lecture Notes in Artificial Intelligence (pp. 576–588). Berlin: Springer-Verlag.

Arcos, J. L. (2004). Improving the quality of solutions in domain evolving environments. In P. Funk & P. A. Conzález-Calero (Eds.), *Proceedings of the 7th European Conference on Case-Based Reasoning*. No. 3155 in Lecture Notes in Artificial Intelligence (pp. 464–475). Berlin: Springer-Verlag.

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 R. Bareiss (Ed.), *Proceedings of the DARPA Case-Based Reasoning Workshop* (pp. 215–224). San Mateo, CA: Morgan Kaufmann.

Cheetham, W., & Price, J. (2004). Measures of solution accuracy in case-based reasoning systems. In P. Funk & P. A. Conzález-Calero (Eds.), *Proceedings of the 7th European Conference on Case-Based Reasoning*. No. 3155 in Lecture Notes in Artificial Intelligence (pp. 106–118). Berlin: Springer-Verlag.

Cheetham, W. (2000). Case-based reasoning with confidence. In E. Blanzieri & L. Portinale (Eds.), *Proceedings of the 5th European Workshop on Case-Based Reasoning*. No. 1898 in Lecture Notes in Artificial Intelligence, (pp.15–25). Berlin: Springer-Verlag.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Cox, M. T., & Ram, A. (1999). Introspective multistrategy learning: On the construction of learning strategies. *Artificial Intelligence*, *112*, 1–55.

Cox, M. T. (2005). Metacognition in computation: A selected research review. *Artificial Intelligence*, *169*(2), 104–141.

Craw, S., Wiratunga, N., & Rowe, R. C. (2006). Learning adaptation knowledge to improve case-based reasoning. *Artificial Intelligence*, *170*, 1175–1192.

Delany, S. J., Cunningham, P., Doyle, D., & Zamolotskikh, A. (2005). Generating estimates of classification confidence for a case-based spam filter. In H. Muñoz-Avila & F. Ricci (Eds.), *Proceedings of the 6th International Conference, on Case-Based Reasoning*. No. 3620 in Lecture Notes in Artificial Intelligence (pp. 177–190). Berlin: Springer-Verlag.

Fox, S., & Leake, D. B. (2001). Introspective reasoning for index refinement in case-based reasoning. *Journal of Experimental & Theoretical Artificial Intelligence*, *13*, 63–88.

Leake, D. B., Kinley, A., & Wilson, D. C. (1995). Learning to improve case adaption by introspective reasoning and CBR. In M. Veloso & A. Aamodt (Eds.), *Proceedings of the First International Conference on Case-Based Reasoning*. No. 1010 in Lecture Notes in Artificial Intelligence (pp. 229–240). Berlin: Springer-Verlag.

Leake, D. B., & Wilson, M. (2008). Extending introspective learning from self-models. In M. T. Cox & A. Raja (Eds.), *Metareasoning: Thinking about thinking, Papers from the AAAI Workshop* (pp. 143–146). Technical Report WS-08-07. Menlo Park, CA: AAAI Press.

Leake, D. B. (1996). Experience, introspection, and expertise: Learning to refine the case-based reasoning process. *Journal of Experimental & Theoretical Artificial Intelligence*, *8*(3), 319–339.

Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., et al. (2005). Retrieval, reuse, revision, and retention in CBR. *Knowledge Engineering Review*, *20*(3), 215–240.

Murdock, J. W., & Goel, A. K. (2008). Meta-case-based reasoning: Self-improvement through self-understanding. *Journal of Experimental & Theoretical Artificial Intelligence*, *20*(1), 1–36.

Wilson, T. D., & Schooler, J. W. (1991). Thinking too much: Introspection can reduce the quality of preferences and decisions. *Journal of Personality and Social Psychology*, *60*(2), 181–192.