CaseAdvisor: Supporting Interactive Problem Solving and Case Base Maintenance for Help Desk Applications

Qiang Yang Edward Kim Kirsti Racine

Case Based Reasoning Group
School of Computing Science
Simon Fraser University
Canada, V5A 1S6
quang@cs.sfu.ca
http://www.cs.sfu.ca/cbr
Voice: 604-291-5415
FAX: 604-291-3045

Abstract

We present CASEADVISOR system for supporting help desk applications. CASEADVISOR improves existing case based reasoning systems on several fronts. Its decision forests help compress a large case base by allowing very similar cases to be merged together. Its maintenance ability allows cases to be compared and redundant cases weeded out. In this article we present the application area and characteristics of CASEADVISOR as well as some of its main advantages over the existing CBR systems.

1 Introduction

Case based reasoning, representing a new generation of expert system technology, relies on the retrieval, reuse, and revision of stored cases. Detailed descriptions of the technology can be found in books written by Kolodner [Kol93] and Leake [Lea96]. Research has been conducted in various subfields of CBR; some references are [HH92, KM92, Sim92, SKR94, SC95, RS93].

In a case based reasoning system, each case is a combination of a problem description and a solution. The case base is organized to allow for efficient real-time retrieval. Once retrieved, a user can adapt the case through simple modifications to solve their problem. In help desk applications, for example, the customer provides information about the particular problems they are encountering. The *customer service representatives* (CSRs) who operate the help desk system, enter the customer's information into the system. Case based reasoning is then employed to retrieve similar problem cases, each with a recommended course of action. Using the courses of action suggested for similar problems as a guideline for problem solving, the CSR can make recommendations to the customer. A distinguishing feature of case based reasoning is the ease with which knowledge can be entered and updated.

CASEADVISOR is an intelligent problem diagnosis and resolution system developed by the CBR group at Simon Fraser University. It allows a help desk organization to capture and reuse the experience and knowledge of its most experienced help desk and customer support personnel in a knowledge database that is easy to build, maintain and use. The system, complete with an OO/C++ API and graphical user interface, is built using intelligent case-based reasoning technology for applications in enterprise knowledge management.

In this paper we present an overview of the CASEADVISOR system and its underlying technological advances in case base maintenance, interactive action planning using decision forests and case adaptation. The system is designed to support a help desk organization in technical problem resolution, knowledge reuse and interactive planning.

2 Overview of the System

In this section we present an overview of the CaseAdvisor system and its application domains. We pay special attention to the underlying domain characteristics which motivate the ongoing development of the case based reasoning system.

2.1 Domain

A typical call center help desk, which is capable of receiving calls from across the country, handles tens of thousands of calls per month. Approximate costs to diagnose and resolve a customer's problem over the telephone may be in the neighborhood of several dollars per call, while the cost of dispatching technical service personnel to a customer site may cost in excess of over 10 times higher.

The high differential between the cost of resolving customer problems over the telephone versus in person provides strong motivation for the development of knowledge-based decision-support systems that can help a service oriented company reduce it's "truck roll rate" as well as improve customer service quality and reduce training costs. By encoding most of the knowledge about the domain in a case based system, suggestions for solutions to customers' problems can be given more effectively in real time. In addition to providing solutions to problems, the system can also provide computerized training appropriate for new technical service representatives and for new services provided by the industry. A critical problem in the help desk industry today is how to provide high-quality customer support at a reasonable cost. Customer service is a knowledge-intensive task, often provided by CSRs who receive varying levels of training and documentation.

Due to these factors, service sector companies may:

- dispatch technical personnel to customer sites to resolve problems they cannot immediately solve over the telephone due to a lack of relevant knowledge or experience by CSR's;
- provide inconsistent answers to customer problems, or unnecessarily transfer customer calls to other personnel due to a lack of experience or knowledge of a particular customer problem area;
- be unable to respond adequately to customer problems in new product areas, such as internet access in Cable-TV industry, at the time of the products' introduction; be unable to quickly access relevant documentation or literature available to resolve particular customer problems;

- lose the knowledge and experience of their most experienced customer service representatives if they choose to leave or pursue another positions within the organization.

2.2 Characteristics of the Domain

Several domain characteristics motivated the development of the CASEADVISOR system. First, the help desk domain is knowledge intensive. Much of the knowledge and experience of a technical service representative is built up over time through the direct experience of working through and trouble-shooting numerous customer problems. A large portion of this knowledge is in the form of electronic user manuals, flow charts and schematic diagrams.

Second, the help desk domain also requires real time interactive response. This task is becoming increasingly difficult because CSRs are usually faced with a high call volume, an increasing variety and complexity of customer problems, and a general need to provide prompt cost effective service to their customers. This is especially true in light of emerging competition in the help-desk industry and the introduction of new services. In the case of cable-TV companies, services such as Internet access and digital music channels are being introduced; likewise, telephone companies are introducing cable-TV services.

Third, today's help desk must be innovative. In telecommunications sales sector, a sales representative must be able to respond in real time to a customer's request regarding a certain purchase. Often, the customer has only a subset of the functional requirements of a target telephony system. The help desk system must be able to determine the feasibility of the product configuration (e.g., do these parts work together?) based on the user's description and system constraints. The recommendations must be offered interactively, by working with the customer through a requirement acquisition and definition process.

2.3 Tasks

Based on the above observations, we concluded that the target case based reasoning system must support the following functional requirements:

- It must provide easy-to-use case maintenance facilities. Due to the fast changing nature of the help desk domain, there must be a mechanism for the system to scan in vast amounts of knowledge in electronic form, and to process this knowledge to isolate the most typical, non-redundant and correct cases and store these cases in its case base.

- It must provide interactive real-time problem solving capability. The CBR system is similar to an AI real-time planning system. A CSR first makes an initial judgment on the problem categorization in order to focus on a problem area. The CSR then solves the problem for a customer by repeatedly listening to a user's description of the problem symptoms, and then suggesting action sequences for the user to follow. If, in the course of the interaction, the CSR determines that the problem area is incorrect, a fast recourse must be provided to backtrack to another problem area to work on.

- It must be capable of working together with a problem solving system such as a constraint-satisfaction system for case adaptation. In particular, the system architecture must be open, and must be flexible enough to provide a mechanism for another specialized problem-solving system to act as a plug-in to the CBR system. One candidate for such plug-in system is a constraint-satisfaction system.

2.4 Benefits of Using CBR Techniques

In the help desk domain, a CBR system satisfying the above functional requirements promises to offer the following advantages. It will:

- reduce the need for expensive technical service calls to customer's homes. Technical service representatives are only sent out to the customer sites when absolutely necessary. Most of the problem are solved on the phone using stored knowledge and experience.

- increase the "consistency" of answers to customers' problems through a standardized knowledge base of solutions and answers to customers' problems, even when the customer talks to two different CSRs at different times.

- reduce the training time for customer service representatives. That is, it helps keep most of the knowledge in the system and makes training more consistent.
- provide immediate upgrade service support in new technical and business services that are introduced at a faster pace than ever.

3 Case Representation and Acquisition

In this section we describe the case authoring and maintenance module of CASEADVISOR in more detail. We also describe our current method for acquiring cases and for "cleaning" the case base.

3.1 Representation and Contents of Cases

The majority of the work in case based reasoning has concentrated on cases with well defined features. These cases have a relational structure, where each feature is more or less a field in a relational database. In reality, however, formulating a case into a structured format requires extensive knowledge engineering. For a given domain, the user has to first determine the important features to use to represent each case. Then a decision has to be made on the type of values for each feature. The process of authoring knowledge in this feature-value format requires extensive maintenance when a new feature is discovered and inserted, or when an existing feature becomes irrelevant.

In industrial practice, a majority of the case bases come directly from either unstructured text documents, which are scanned in, or end-users' verbal description. These cases may have generic features such as *problem description* and *problem solution*, but each of these features probably will not be further partitioned down to a relational level. As an example, in a computer-printer repair domain, a case might be described as:

```
Problem Description:  Paper continues jamming laser printer due to dirty
and/or sticky internals.

Solution :  The internal components of the laser printer are dirty and
perhaps gummed up.  There is also a possibility the paper is sticking together.
Running regular gummed labels through a laser printer is a key source of
the problem because the high heat melts the gum labels.
```

Structured, relational case features often facilitate maintenance. Each attribute is associated with a set of values. The cases can be scanned and values that appear infrequently for a particular attribute can be modified or brought to the user's attention. Alternatively, integrity constraints can be specified ensuring that each value entered is a legal one for that attribute. Unstructured cases, on the other hand, are more problematic. Often the cases can not be reduced to a set of attribute-value pairs so that even range checking can be a complex problem.

The case-base management components of the CASEADVISOR system, known as CASEAUTHOR and CASEMAINTAINER are able to account for both unstructured cases as well as structured cases: they allow the users to enter these cases with the ease of a word processor, and they perform case comparisons to weed out redundant and incorrect cases. The representation of a case is as follows:

Keywords Keywords are short descriptions of the case which can be used in fuzzy string matching with the users initial free-form English text input.

Questions These are the features in a typical case represented in relational form. The questions with multiple choice answers provide an indexing mechanism for logarithmic time retrieval of the cases.

Case Description This is a more detailed textual description of the case used to confirm the general problem area. A CSR will use this description to confirm that the problem being diagnosed indeed falls into the intended problem category. Multimedia representation is supported for this field, where a hyper link can be provided using HTML to any HTML compatible image, video or audio data format.

Case Solution The case solution provides a solution to the case in either textual format or any multimedia format.

A second component of the case solution is a decision tree. The tree can be attached to the case to provide further diagnosis and interactive action planning. For example, once a CSR has determined that a user is experiencing

a VCR problem through initial interactive diagnosis, a more detailed diagnostic and repairing session may be initiated. In this latter session, a decision tree repeatedly prompts the CSR for the next action to perform and asks a follow-up question. In the VCR case, an action such as "run cable directly through the TV bypassing the VCR" can be suggested. A follow-up question might be: "Now what does the screen look like?"

In addition to the text and multimedia data format, the case solution can also contain additional data structures which suggest an initial solution to the problem for an adaptation system to work on. For example, a case containing a product configuration holds the different product components (for example, computer hardware parts) and constraints (compatibility between a memory chip and the motherboard).

The case structure described above has been used to provide solutions in a variety of problem domains. Some examples are:

- In the Cable-TV call center domain, the cases provide problem area descriptions, general solutions and detailed solutions to a technical problem. The solutions are presented in various multimedia format.
- In a software reuse domain, the cases are used to represent past best practices of software feasibility studies. These studies are used as guidelines for composing future software contracts.
- In a computer-language tutoring domain, the cases provide confirmation to a problem area in which a student is interested, and a corresponding solution with an associated example to assist the student.
- In a mutual-fund advisory domain, the cases provide prompts for a user to formulate financial goals in the user's language and recommendations for the type of mutual fund in which the user might be most interested.
- In a sales advisor domain for selling computer hardware, the cases provide high-level functional requirements for the user to specify. The solutions contain both a model system package and a set of constraints for the user to use in tweaking the model system. A user can obtain what-if information from the CBR system in real time.

3.2 How to Acquire Cases

Cases are acquired in three ways, through a word processor-like case authoring component (CASEAUTHOR), a case-retrieving agent (CASEMAINTAINER) and an online unresolved problem-case acquisition functionality.

Case authoring through CASEADVISOR CASEAUTHOR Case authoring in CASEADVISOR is a simple matter of typing in a case title, a short description of the case in plain English text and associating questions and solutions with the case. The case authoring features advanced graphical, drag-and-drop based user interface which allows a user to type in case contents and manages the cases in a natural fashion. A user can also use a graphical user interface to specify the importance of a question-answer pair to a case, measured in the form of weights.

Case maintenance through CASEADVISOR CASEMAINTAINER With CASEMAINTAINER one can scan in documents in textual form and convert them to cases. The CASEMAINTAINER subsystem processes these documents using information retrieval techniques, and suggests questions and keywords to be attached to the cases. The maintainer also checks redundancies and inconsistencies in a case base. When two cases are solving the same problem, the more powerful solution is retained and the less useful one is presented to the user who maintains the case base. With the user's confirmation, the subsumed case will be discarded. Finally, a maintainer allows users to merge two case bases together while ruling out repetitive cases.

With a large legacy case base, a need arises to detect if two cases are equal or if one case subsumes another by some criteria determined by the background knowledge. A special case is when two cases are considered equivalent; that is, all attribute values are identical.

An example of redundancy in the printer repair domain is displayed in Table 1. It demonstrates the difficulty of identifying redundant cases when the cases are unstructured. A fuzzy string comparison of the two cases presented will detect some similarities, but there are significant differences between the cases.

Our approach to solving the maintenance problem for unstructured case bases is to integrate an agent within a case based reasoning system. In order to minimize the knowledge acquisition bottleneck, the agent allows unstructured cases to be processed as well as the structured ones. We first use an information-retrieval based algorithm to parse

Case 1
CASE NAME: Envelopes jam laser printer due to glue. SOLUTION: Normal envelopes and laser printers do not get along together. Problems include poor glue heat tolerance.
Case 2
CASE NAME: Paper continues jamming printer due to sticky internals SOLUTION: Envelopes do not work very well with laser printers. The high heat melts the gummed labels.

Table 1: Example of redundant cases in the printer repair domain

the cases by mining key words and key phrases from the unstructured text. These key words and phrases will offer the basis on which subsequent modules can operate. After the information retrieval step, we then use a specialized redundancy detection and inconsistency detection module to manage the case base.

Once the keywords are extracted, we can use a set of general algebraic rules to detect redundancy in a case base. As an example, one rule states that:

Case 1 : Problems (p_1, p_2) Solution (s_1)

Case 2 : Problems (p_1, p_2, q_1) Solution (s_1)

q_1 can either be a keyword or a set of words containing a keyword. In this case, Case 1 subsumes Case 2. The sufficient conditions for solution s_1 have been established to be (p_1, p_2). The value of q_1 is irrelevant. Once the first two premises hold, the solution can be offered to the user. Once this scenario has been detected, the system allows the user to view both cases and highlights the unnecessary condition. As it is possible that Case 1 is an inconsistent case, the fact that it subsumes Case 2 does not mean that Case 2 should be summarily deleted from the case base. The user must examine both cases and decide on the suitable course of action.

Unresolved Problem-Case Functionality The CASEADVISOR system can also react gracefully to the arrival of new cases. If a problem description results in no credible cases being retrieved, or if the cases retrieved do not solve the problem, a user of the case base can author a template to be stored on disk, which can then be retrieved by a user maintaining the cases as a basis for authoring a new case.

3.3 Learning Mechanisms

Learning in CASEADVISOR is achieved in two levels, the case level and the weights level. At the case level, the system can acquire new cases from users through the unresolved problem-case functionality described above, and can periodically "clean" a case base to obtain a more efficient and up-to-date one using the above information retrieval and algebraic redundancy detection methods.

A second level of learning exists in the similarity computation. The similarity measurement between cases is accomplished through the use of weights that are attached to the question-answer and case pairs. These weights can be specified initially by the user. In our on-going work, we are developing a neural network based feedback mechanism which, based on the user's behavior in case retrieval, can automatically adjust the weights of used in similarity computation.

4 Use of Cases

In this section we discuss how the cases are used in the context of help desk environments. We pay special attention to the storage and retrieval of cases as well as the actual use of the cases by the end user.

4.1 How to Use Cases

Cases are used by the CSRs in two ways. One is to provide a solution guideline in solving a problem. In a typical scenario, a customer representative receives a phone call from a user who reports a problem. In response, the CSR enters a short description of the problem in English. The system replies by returning a set of likely cases associated with questions. The responses to these questions are used to further narrow down the retrieved cases.

At this point, a user can open a case up to look at its more detailed descriptions. Part of the description is a problem repair plan in the form of a sequence of actions. Between each pair of adjacent actions is a confirmation question, with an appropriate answer to which the the next step in the plan would make sense. Multimedia information provides cues for the user to solve the target problem. Thus, the first use of the system is through its function as an advisor. This is used most often in Cable-TV and telephony call center environments. The second use of the cases is to provide a basis for the user to further perform reasoning and analysis. In a sales advisor environment, the system retrieves the case that most closely matches a user's initial description. The description is a product model that can be further tweaked. For example, a user working on a computer systems advisor might prefer to buy a 17" monitor instead of a 21" one. The system will then advise the user of the impact of that decision on the system performance and the overall cost. Furthermore, any infeasible changes will not be allowed.

4.2 Indexing and Similarity Computation

The CASEADVISOR system uses an indexing mechanism aimed at allowing efficient scaling of the case base. Cases are indexed on problem features, which are the questions authored by the CSRs. Once an English description of the problem is received, the system retrieves the cases that most likely match important keywords in the description and ranks the cases. The top n high-ranking cases are retrieved. When n is very large, the user can then choose to answer some questions associated with these cases to narrow down the search. The questions serve as a logarithmic indexing structure which will dynamically re-rank all retrieved cases. Similarity computation will proceed using a nearest neighbor formula, which is based on the question-case weights and the answers to the questions (which correspond to the values of features in a structured case base). The weights are not fixed for an application domain; each case have different weights for answering the same question.

4.3 Adaptation

The CASEADVISOR system comes with an application programming interface (API). Through the API we have built an adaptation system for sales configuration, in a typical help desk environment.

Consider the example of buying a new computer system. Suppose, for simplicity, this requires that you buy a monitor, a printer, and the computer (the "box"). The most basic constraint is that the parts should work together (i.e. if brand M monitor is incompatible with brand B box, then such a configuration should not be allowed). Furthermore, *requirements* should also be automatically specified: if a particular printer requires a special cable, then it is important to know that this cable is needed to make the system work.

Along with basic compatibility and requirement constraints, there are user imposed constraints. For example, the user may want the system to cost less than $3000 in total, or that the printer be a laser printer, or that the system be powerful enough to handle a particular application.

Using CASEADVISOR API, we store a set of model product configurations for computer equipments as cases. A user can interactively specify the functional requirements of the target system. For example, the user can choose an answer to the question "Do you want access to the Internet?" Based on the answers, the CASEADVISOR system will retrieve a ranked list of product configurations that are recommended to the user. This is where adaptation can be applied. If the user is not satisfied with a retrieved case, and wishes to modify the system by replacing a certain component (a monitor type, for example), the user can then activate a constraint-satisfaction system which performs a local search. During local search all constraints of the sales configuration domain are checked and as many constraints as possible are satisfied. The system returns an adapted case within a pre-specified time limit.

4.4 Integration With Other Reasoning Mechanisms

One of the strengths of the CASEADVISOR system is its integration with decision tree technology. It is widely acknowlodged that decision trees provide a competing technology to CBR, because of its structured means of retrieving

and organizing information. One major shortcoming of decision trees is that system questions must be posted in a sequence before a final answer is reached. If any answer is missing, the system will not be able to continue. As a result, many knowledge base applications have chosen to use CBR technology instead.

However, through extensive customer interaction we discovered that many customers prefer to have limited decision tree functionality. For example, in a call center enviroment, a CSR can use a CBR system to bring the problem scope down to a single area. At this point, the CSR repeatedly asks the user questions and suggests actions. The actions suggested depend on the answers to the questions asked. This user model suggests a mixed CBR-decision tree system.

Our answer is to maintain a decision forest. For each problem case the solution consists of two parts. The first part is a piece of general advice on the solution to the problem. If the user chooses, he/she can also check out the second part of the case, which is a decision tree. The CSR follows the tree in an interactive manner. In this way, a more experienced CSR can quickly solve the problem based on the general knowledge, and a less experienced one can rely on the more detailed knowledge provided by the decision trees to solve problems.

As an example, a CSR in a Cable-TV domain might use a case base to narrow down the search to VCR reception problems. A general solution in this problem area is to turn the VCR on and off. A more detailed solution which follows the first action will depend on the user's answer to the question "does the problem occur on all channels or some channels?" Each option leads to a distinct sequence of actions to pursue.

Another strength of the CASEADVISOR system is its ability to be integrated with a rule base. Through a rule base the system can infer the answers to questions based on previous answers and keywords. This helps reduce the total time in finding a solution case. For example, if the problem description " I have VCR recording problems" is entered, then the question "What kind of problems are you experiencing?" is answered automatically. Users will find it annoying to answer these questions again.

4.5 How to Present the Result to Users

The CASEADVISOR user interface is in fact "internet ready", by displaying case contents through the Netscape browser. By using browser technology we can display any information that can be displayed on the Web. Another advantage is that we no longer need to train the user in using the interface system, as most users of the CASEADVISOR system are already familiar with the web technology.

Figure 1 shows an example of how cases are presented to the user through Netscape browser.

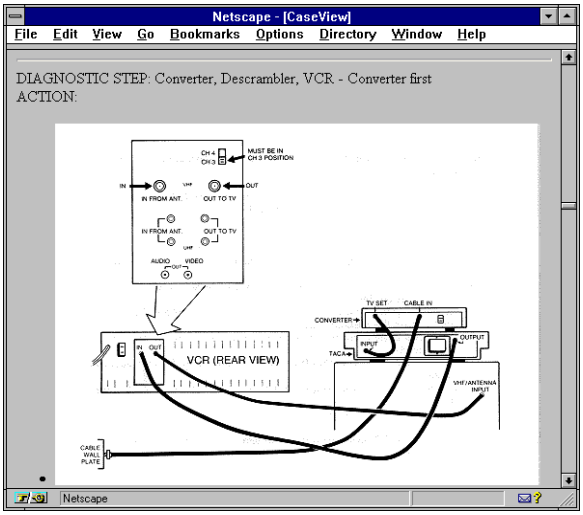

Figure 1: A display of case information in Netscape.

5 Evaluation

The CASEADVISOR system is undergoing extensive beta-testing at several customer sites. The empirical testing consists of usability testing in call center environments, empirical testing involving selected users, and comparing with other systems.

5.1 Empirical Examination

One component that has undergone extensive user testing is the CASEAUTHOR component. End users' feedback on the current version of the system has indiciated that it is as easy to use as a PC based word processor. The problem resolution component of the system satisfies the real time requirements in call centers.

Another component of the system that has undergone extensive testing is the CASEMAINTAINER subsystem. We now present part of the testing results. The redundancy-detection module in CASEMAINTAINER is responsible for testing to see if an incoming case produces possible redundancy. If there is no possible redundancy, the case is simply added to the existing case base. If there is, the case is presented to the user along with the case causing the possible conflict. The user then determines which, if any, of the cases should be deleted from the case base.

5.2 Comparison With Other Systems

Most commercial CBR shells provide both case authoring and problem resolution facilities. Examples of well known systems are Inference's CBR Express and CBR2/3 and ISoft's ReCall and others. Haley enterprises EasyReasoner allows users to organize information from a database in the form of decision trees. However, we are aware of no other systems that have all three main features as in the CASEADVISOR system. These three features are:

- case base maintenance facility to find redundant and incorrect cases,
- the ability to integrate decision forests with cases seamlessly, allowing very similar cases to be merged together;
- the ability to combine CBR with a specialized reasoning system such as constraint satisfaction in an easy manner.

In the CBR research community, several proposals have been presented for dealing with case base maintenance. David Aha [Aha91] presented several case-based learning (CBL) algorithms which were tolerant of noise and irrelevant features. Smyth and Keane [SK95] suggested a competence-preserving deletion approach. But none of these algorithms dealt specifically with unstructured case representations and the problem of how to detect redundant cases.

6 Other Characteristics

6.1 CBR under Internet environment

The CASEADVISOR system includes a Web server subsystem[1] which is capable of displaying the problem resolution component on the internet. Using this system a end user can have access to on-line help and problem resolution 24 hours a day.

7 Conclusions

In this paper we have presented an overview of the CASEADVISOR system and its application in the call center environment. We have described several key technological advances represented by CASEADVISOR, including case maintenance, interactive problem solving with decision forests and case adaptation in sales configuration. In the future, we plan to expand CASEADVISOR's capabilities in advanced weight and similarity learning, integration of Bayesian networks with case based reasoning, and more structured organizations of very large case bases.

[1] http://www.cs.sfu.ca/cbr

Acknowledgment

We thank NSERC, BC ASI, BC Science Council and Canadian Cable Labs Fund for their support of the CASEADVISOR project.

References

[Aha91] D. Aha. Case-based learning algorithms. *Proceedings of the 1991 DARPA Case-Based Reasoning Workshop*, 1, 1991.

[HH92] D. Hennessy and D. Hinkle. Applying case-based reasoning to autoclave loading. *IEEE Expert*, 7(5):21–27, 1992.

[KM92] J. Kolodner and W. Mark. Case-based reasoning. *IEEE Expert*, 7(2):5–6, 1992.

[Kol93] Janet Kolodner. *Case-based Reasoning*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1993.

[Lea96] David Leake. *Case-based Reasoning – Expriences, Lessons and Future Directions*. AAAI Press/ The MIT Press, 1996.

[RS93] A. Ram and J.C. Santamaria. Continuous case-based reasoning. *Proceedings of the AAAI-93 Workshop on Case-Based REasoning*, pages 86–93, 1993.

[SC95] B. Smyth and P. Cunningham. A comparison of incremental case-based reasoning and inductive learning. *Proceedings of the 2nd European Workshop on Case-Based Reasoning*, 2, 1995.

[Sim92] E. Simoudis. Using case-based retrieval for customer technical support. *IEEE Expert*, 7(5):7–13, 1992.

[SK95] B. Smythe and M. Keane. Remembering to forget : A competence-preserving case deletion policy for case-based reasoning systems. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, volume 1, Montreal, Canada, August 1995.

[SKR94] R. Shank, A. Kass, and C. Riesbeck. *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, New Jersey, 1994.