The application of case properties in maintaining case–based reasoning systems

Ioannis Iglezakis[1], Thomas Roth–Berghofer[2], and Christina E. Anderson[3]

[1] DaimlerChrysler AG, Research & Technology, FT3/AD,
P.O. Box 2360, 89013 Ulm, Germany
ioannis.iglezakis@daimlerchrysler.com

[2] tec:inno GmbH,
Sauerwiesen 2, 67661 Kaiserslautern, Germany
thomas.roth-berghofer@tecinno.com

[3] University of Kaiserslautern,
P.O. Box 3049, 67653 Kaiserslautern, Germany
anderson@mathematik.uni-kl.de

Abstract. The role of maintenance in the domain of case–based reasoning systems is extremely important. Therefore, methods of maintaining case bases using case properties will be presented. Definitions of the case properties (correctness, consistency, incoherence, minimality, and uniqueness) are given. The use of these properties in five experiments is explained, and the results of these experiments on nine real world case bases is discussed. These results can also be viewed in various tables that are included for clarification.

1 Introduction

Today case–based reasoning technology can be found in many knowledge management systems. And the longer these systems are running the greater is the demand for maintaining these systems. This demand is widely acknowledged and maintenance for case–based reasoning systems is a hot topic in current research. In particular, the maintenance of the case base has received much attention [7]. Therefore, we also focus our attention on case base maintenance (CBM) activities that affect the case base rather than the vocabulary, similarity, or adaptation knowledge containers [10].

Case base properties and quality measures have been defined to assess the quality of the case base, and operators have been described to remedy encountered deficiencies. First tests and evaluations showed the usefulness of the measures and operators but a detailed analysis was missing. In this paper, we describe experimental procedures and present results in detail.

This paper is organized as follows: in the next section, we recapitulate the case and case base properties as well as the two additional steps of the six step process review and restore defined in [9]. We then describe our experimental design and the results. We close the paper with concluding remarks and issues for future work.

2 The Case Properties

The first substep in the extended case–based reasoning cycle is to measure the case base through case properties. These measures are used as an indicator of quality within a case or between groups of cases. Formal definitions for the case properties *correctness*, *consistency*, *uniqueness*, *minimality*, and *incoherence* can be found in [8, 9]. In the following sections only informal explanations will be given.

2.1 Correctness

Correctness is the only isolated property. The reason is that the correctness of a case can be realized without any other case. Each of the other properties involve the given case as compared with other cases. For the following applications correctness is the only isolated property that must be considered. A case is considered correct if and only if the given solution component actually solves the problem that is specified by the problem component. Note that correctness is assumed in the experiments that are described below.

2.2 Consistency

Consistency is the first of the comparative case properties. A given case is called consistent if and only if there is no other case or a subset of the given case in which the attributes and their values are the same but for which the solutions are different. To illustrate this case property a counter–example from the help–desk domain is shown in table 1.

Table 1. Example for (in-) consistency

Operating System	Printer	Printing	Paper	Ink	Solution
WinNT	HP850c	No	Yes	Yes	*Update Driver*
WinNT	HP850c	No	—	—	*Second Level*

In this example, the two cases have the same operating system and the same printer, and in both cases there is no printing. However, the first case contains more information than the second. Regardless of this difference in the number of attribute–value pairs, the solutions are not the same, and therefore the second case is inconsistent.

2.3 Uniqueness

Uniqueness is also a comparative property. A given case is called unique if and only if there is no other case which has the same attribute–value pairs and solution. Finding a counter–example is not difficult. Again the help–desk domain is used for illustration and the example can be viewed in table 2.

Table 2. Example for (not) unique

Operating System	Printer	Printing	Paper	Ink	Solution
WinNT	HP850c	No	Yes	Yes	Update Driver
WinNT	HP850c	No	Yes	Yes	Update Driver

For the two cases in this example, each attribute–value pair in the first case is identical to the corresponding pair in the second case. Furthermore their solutions are also the same, hence the two cases are obviously not unique.

2.4 Minimality

Minimality is the opposite of subsumption. A given case is called minimal if and only if there is no other case with the same solution and whose attribute–value pairs are a real subset of the given case. Table 3 contains a counter–example to clarify this concept.

Table 3. Example for (not) minimal

Operating System	Printer	Printing	Paper	Ink	Solution
WinNT	HP850c	No	Yes	Yes	Update Driver
—	HP850c	No	Yes	Yes	Update Driver

The two cases in this table are identical except that the second case has no value given for the operating system. This means that the second case's attribute–value pairs are a real subset of the first case's attribute–value pairs. Therefore, the first case is not minimal.

2.5 Incoherence

Finally, the last comparative case property is incoherence. A given case is incoherent if and only if there are no other cases with the same solution and for which the attribute–value pairs of the two cases are equal with the exception of a (small) specific number (Δ) of attributes. Once again, a counter–example illustrates this case property (cf. table 4).

The two cases in table 4 differ only in the attribute operating system. Because the solutions are the same and with the assumption that $\Delta = 1$, they are equal in every attribute with the exception of one (Operating System). Hence the two cases are not incoherent.

In this section the notations needed to define the case properties were given. Then these properties were introduced in conjunction with examples from the help–desk domain. These definitions are the basis for the case base properties, in which the case

Table 4. Example for (not) incoherent (for $\Delta = 1$)

Operating System	Printer	Printing	Paper	Ink	Solution
WinNT	HP850c	No	Yes	Yes	Update Driver
Win95	HP850c	No	Yes	Yes	Update Driver

properties are applied to the entire case base. For example, if every case in the case base is consistent, then the entire case base is also consistent by definition. To obtain quality measures, the case properties are measured in degrees within the case base. These degrees offer a sophisticated method of measuring the case base. The aim of these degrees is the mapping of sets to numbers. That means that they divide the number of "good" cases by the total number of cases in the case base. In [8, 9] more information on quality measures and degrees of case base properties can be found. In the following section, the experimental procedure describes how to use the case and case base properties in an attempt to minimize the size and maximize the accuracy of the case base.

3 Experimental Procedure

Through the use of examples the definitions for the components used in building case and case base properties have now been explained and clarified. In this section one possible method of application is presented. It consists of a series of experiments used on the case base. Preferably without decreasing the accuracy, the case properties determine which cases are added and deleted to the case base as a means of optimizing the case base. The benchmark result can be seen as the baseline algorithm and each experiment is described in relation to this benchmark result that is achieved without the use of the case properties. The explanation of these experiments is followed by actual results.

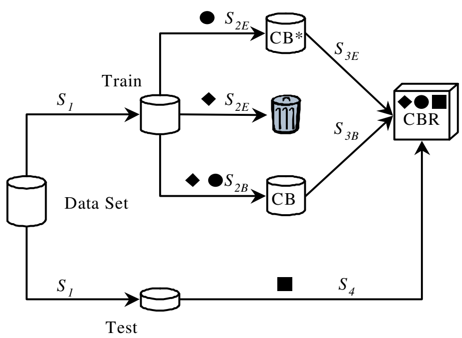

Fig. 1. Benchmark/Experiment 1

To reach the first benchmark result the data set of cases is divided into a training set and a test set (S_1). The training set becomes the case base (S_{2B}). The training set is applied to the case–based reasoning algorithm (S_{3B}), and then each case from the test set is also tested against the case–based reasoning algorithm (S_4). A certain accuracy and case base size are obtained. For experiment one the training case base is optimized (S_{2E}). In the process of optimization, the first case is used to start the case base. Then the second case is checked against the first case. If there are any contradictions in relation to the used case property (for example, if uniqueness is used as the case property and the two cases are identical, then uniqueness is violated), then the case inside the case base which contradicts the newly added case is removed. In our example, the first case would be thrown away if the two cases were identical. Therefore, experiment one is accomplished by optimizing the case base before applying it to the case–based reasoning algorithm (S_{3E}). Then the test cases are applied and a level of accuracy is reached (S_4).

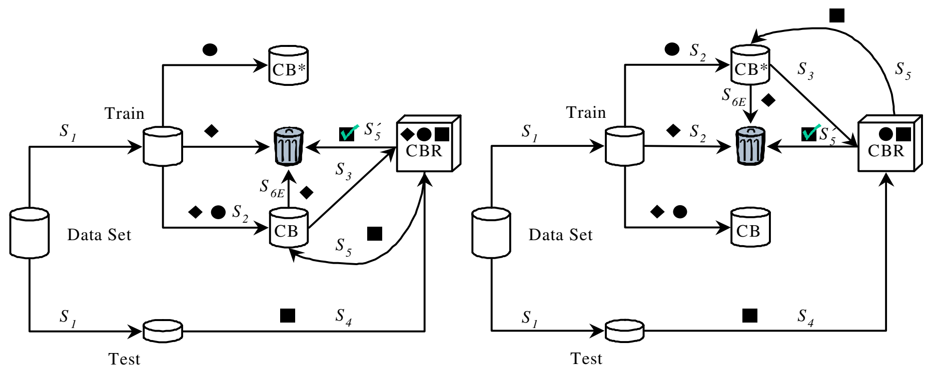

Fig. 2. Benchmark/Experiment 2,3 on the left side, and 4,5 on the right side

The second benchmark result is very similar to the first (S_1, S_2, S_3, and S_4). The only difference is that the tested case, after being applied to the algorithm, is added to the case base (S_5). Here also a certain accuracy and case base size are obtained. In addition, experiment two removes each case from the case base which contradicts the property used for the newly added test case (S_{6E}). Again the accuracy and case base size are realized.

The third benchmark result is an extension of the second and similar to the CBL2 algorithm described in [1]. Cases which are predicted correctly by the case–based reasoning algorithm do not find their way into the case base (S'_5). Only cases which receive a wrong prediction are added to the case base because their knowledge is needed for further prediction usage (S_5). Experiment three then removes again every case from the case base which contradicts the property used for the newly added test case (S_{6E}). Finally the accuracy and case base size are achieved.

The fourth benchmark result starts now with the optimized[1] case base (S_1, S_2, S_3, and S_4). Then the test cases are added to the case base after being applied to the case–based reasoning algorithm (S_5). At the end the accuracy and case base size are attained. Experiment four uses the optimized case base too before the application of the case–based reasoning algorithm. However, it also optimizes the case base a second time in relation to the cases that are added at the end of the cycle (S_{6E}). This means that after the test cases have been applied to the algorithm and are being added to the case, the optimization is applied. This guarantees that no cases within the case base contradict the newly added cases. Then the accuracy and case base size are acquired.

Finally, the fifth benchmark result is an extension of the fourth benchmark and is quite similar to the third benchmark. It uses the optimized training case base and adds only cases whose predictions are false to the training case base (S_5'). In conclusion, the accuracy and case base size are obtained. Experiment five extends benchmark five because those cases are removed from the case base which contradict the property used for the newly added test case (S_{6E}). At the end the accuracy and case base size are realized.

Each experiment is implemented and compared to the corresponding benchmark. This gives the observer the opportunity to compare the results of the applications with and without the use of the case properties. Actual results of these benchmarks and experiments are presented below.

4 Experimental Results

The goal of using the predefined benchmarks and experiments described in section 3 is to observe the accuracy and case base size in applications which exclude and include optimization, respectively. In these experiments, the optimization takes the form of maintenance. The detection of case properties in the case base allows the information to be used to optimize the case base in relation to both accuracy and case base size.

In order to conduct these experiments, data in the form of case bases was obtained from the UCI Machine Learning Repository [5]. The following case bases are used: *Annealing* (797 cases), *Audiology* (200 cases), *Australian* (690 cases), *Credit Screening* (690 cases), *Housing* (506 cases), *Pima* (768 cases), *Soybean* (307 cases), *Voting Records* (435 cases), and *Zoo* (101 cases). Each of the following benchmarks and experiments was implemented on each of these case bases to attain the results that are presented below. Therefore the original case bases were split into five folds to fulfill a five–fold cross validation. The case–based reasoning algorithm used was a basic optimistic nearest neighbor algorithm.

The average case base property degree of the five training folds is summarized in table 5. This table shows that the case base properties are able to indicate problems in the case bases. The single values vary across the different data characteristics but usually indicate quality deficiencies for at least two of the case properties. Commonly the highest values are performed for the consistency degree while the lowest values are usually observed for the incoherence degree. An outlier is the Voting Records domain

[1] see experiment one

Table 5. Degrees of Case Base Properties

case base	degree of consistency	degree of incoherence	degree of minimality	degree of uniqueness
Annealing	0.951	0.234	0.779	0.822
Audiology	1.000	0.771	0.975	0.825
Australian	0.995	0.832	1.000	0.988
Credit S.	0.995	0.831	0.999	0.989
Housing	0.997	0.931	1.000	0.983
Pima	1.000	0.864	1.000	0.992
Soybean	0.995	0.743	0.998	0.980
Voting R.	0.451	0.446	0.233	0.717
Zoo	1.000	0.173	1.000	0.443

which has the lowest consistency and minimality degree. This points to major problems within the case base. Indeed after removing two cases which have basically unspecified values in the case representation and contradictory solutions, the consistency degree increases to 0.960 and the minimality degree increases to 0.415. Further utilizations are shown in table 6 – 10.

Table 6. Benchmark/Experiment 1

	benchmark		consistency		incoherence		minimality		uniqueness	
case base	size	acc.	size	acc.	size	acc.	size	acc.	size	acc.
Annealing	637.60	0.97	625.20	0.97	323.40	0.97	551.20	0.97	574.80	0.97
Audiology	160.00	0.70	160.00	0.70	138.20	0.71	156.60	0.70	145.20	0.69
Australian	552.00	0.81	550.60	0.81	503.40	0.79	552.00	0.81	548.60	0.81
Credit S.	552.00	0.79	550.60	0.80	498.80	0.79	551.40	0.79	549.00	0.79
Housing	404.80	0.31	404.20	0.31	390.20	0.30	404.80	0.31	401.40	0.30
Pima	614.40	0.65	614.40	0.65	569.80	0.65	614.40	0.65	612.00	0.65
Soybean	245.60	0.92	244.40	0.92	212.80	0.91	245.00	0.92	243.20	0.92
Voting R.	348.00	0.93	216.40	0.91	191.20	0.92	119.20	0.91	280.60	0.93
Zoo	80.80	0.95	80.80	0.95	30.40	0.92	80.80	0.95	50.40	0.95

Table 6 shows that optimization through the case properties usually reduces the case base size and preserves the prediction accuracy. Furthermore, the comparison between table 6 and table 5 shows that the potential of optimization indicated by the values in table 5 is related to the values of case base properties in table 6. Hence, the order of extension for the proposed quality measures is also appropriate to predict the number of cases that can be saved without losing classification quality.

Table 7. Benchmark/Experiment 2

	benchmark		consistency		incoherence		minimality		uniqueness	
case base	size	acc.	size	acc.	size	acc.	size	acc.	size	acc.
Annealing	797.00	0.97	787.40	0.97	536.20	0.97	740.00	0.97	756.00	0.97
Audiology	200.00	0.72	200.00	0.72	185.80	0.71	198.60	0.72	190.20	0.72
Australian	690.00	0.81	689.40	0.81	656.80	0.80	690.00	0.81	688.40	0.81
Credit S.	690.00	0.80	689.40	0.80	657.00	0.80	689.80	0.80	688.00	0.80
Housing	506.00	0.31	505.60	0.31	496.60	0.31	506.00	0.31	504.40	0.31
Pima	768.00	0.66	768.00	0.66	739.00	0.66	768.00	0.66	766.40	0.66
Soybean	307.00	0.91	306.60	0.91	286.00	0.91	306.80	0.91	305.40	0.91
Voting R.	435.00	0.93	372.40	0.90	271.80	0.93	279.40	0.91	376.00	0.93
Zoo	101.00	0.95	101.00	0.95	47.80	0.96	101.00	0.95	75.00	0.95

Benchmark/Experiment 2 (BE2) to Benchmark/Experiment 5 (BE5) allow the addition of cases, unlike Benchmark/Experiment 1 (BE1). By comparing BE2 with BE3 it can be seen that the accuracy stays mainly the same while the case base size in BE2 is larger than in BE3. This is clear because there is no optimization in the beginning and BE2 is continuously adding cases while BE3 is adding cases selectively. However, the ratio of benchmark to experiment is in BE2 mainly lower than in BE3. The reason is that with continuously adding cases the propability of finding cases in the case base which contradict the newly added cases increases with the size of the case base.

The accuracy between BE2 and BE4 is again quite the same while the sizes and ratios between benchmark and experiment of the case bases used in BE4 are smaller. This is convincing because the case base of BE4 is optimized before it is applied to the case–based reasoning algorithm.

The comparison between BE3 and BE4 is different. While the accuracy is again nearly equal, both case compression techniques (case properties and using the predicted value) play a role. With help of table 5 it is possible to predict in which instance the use of case properties yields to smaller case bases and when the use of the predicted value leads to smaller case bases. It appears that using case properties gives better results, when the degree is around 0.75 and worse otherwise.

Finally, using both compression methods (cf. table 10) gives the best result with a value of accuracy nearly identical to the accuracy which was found in the other Benchmark/Experiments.

In the field of maintaining case–based reasoning systems case properties proved to be promising as indicators for maintenance. This was not only shown in a theoretical way but also in nine real world case bases. These experiments tested new methods of maintenance in the realm of maintaining case–based reasoning systems; they also serve as a first step to encourage new ideas that could further encourage the use of case bases.

Table 8. Benchmark/Experiment 3

case base	benchmark size	benchmark acc.	consistency size	consistency acc.	incoherence size	incoherence acc.	minimality size	minimality acc.	uniqueness size	uniqueness acc.
Annealing	641.80	0.97	637.80	0.97	638.60	0.97	641.80	0.97	641.80	0.97
Audiology	171.60	0.71	171.60	0.71	171.20	0.71	171.60	0.71	171.60	0.71
Australian	578.80	0.81	578.20	0.81	577.00	0.80	578.80	0.81	578.80	0.81
Credit S.	580.00	0.80	579.60	0.80	577.00	0.80	580.00	0.80	580.00	0.80
Housing	474.60	0.31	474.20	0.31	473.20	0.31	474.60	0.31	474.60	0.31
Pima	667.20	0.66	667.20	0.66	666.60	0.66	667.20	0.66	667.20	0.66
Soybean	251.00	0.91	250.60	0.91	251.00	0.91	251.00	0.91	251.00	0.91
Voting R.	354.40	0.93	309.20	0.90	354.00	0.93	328.80	0.90	354.40	0.93
Zoo	81.80	0.95	81.80	0.95	81.40	0.95	81.80	0.95	81.80	0.95

Table 9. Benchmark/Experiment 4

case base	benchmark size	benchmark acc.	consistency size	consistency acc.	incoherence size	incoherence acc.	minimality size	minimality acc.	uniqueness size	uniqueness acc.
Annealing	797.00	0.97	777.20	0.97	352.00	0.97	674.00	0.97	703.00	0.97
Audiology	200.00	0.72	200.00	0.72	169.00	0.72	195.60	0.72	177.00	0.72
Australian	690.00	0.81	688.00	0.81	616.20	0.79	690.00	0.81	685.00	0.81
Credit S.	690.00	0.80	688.00	0.80	610.80	0.80	689.20	0.80	685.00	0.80
Housing	506.00	0.31	505.00	0.31	483.40	0.30	506.00	0.31	501.00	0.31
Pima	768.00	0.66	768.00	0.66	700.40	0.65	768.00	0.66	764.00	0.66
Soybean	307.00	0.91	305.40	0.91	257.00	0.91	306.20	0.91	303.00	0.91
Voting R.	435.00	0.93	250.20	0.89	231.80	0.91	146.60	0.89	342.00	0.93
Zoo	101.00	0.95	101.00	0.95	32.80	0.94	101.00	0.95	59.00	0.95

5 Conclusions and further work

The work presented in this paper is a continuation of work that was begun by several researchers in the field of maintaining case base reasoning systems.

Aha [1] presented four algorithms for case–based learning (CBL). Each of these was designed to complete a learning application involving some sort of problems. The algorithms CBL2, CBL3, and CBL4 reduce the costs of case retrieval by implementing their case addition strategies, while CBL1 worked as a benchmark. Furthermore, Aha [2] and Aha and Breslow [3, 4] worked to refine conversational case libraries (CCL) in order to meet design guidelines of tool manufacturers, which determined how the libraries should be built. In this case they used optimization in the form of a filtering algorithm which reduces the case base size.

Smyth and Keane [11] and Zhu and Yang [13] have found a beneficial way to maintain the case base by using the size of the case base as a trigger to preserve its competence. The objective here is to preserve the performance level of the case base while

Table 10. Benchmark/Experiment 5

case base	benchmark size	benchmark acc.	consistency size	consistency acc.	incoherence size	incoherence acc.	minimality size	minimality acc.	uniqueness size	uniqueness acc.
Annealing	641.80	0.97	626.40	0.97	327.00	0.97	555.40	0.97	579.00	0.97
Audiology	171.60	0.71	171.60	0.71	149.00	0.72	168.20	0.71	157.00	0.71
Australian	578.80	0.81	577.20	0.81	530.60	0.79	578.80	0.81	575.60	0.80
Credit S.	580.00	0.80	577.80	0.80	525.40	0.79	579.40	0.80	577.20	0.79
Housing	474.60	0.31	473.60	0.31	459.60	0.30	474.60	0.31	471.60	0.31
Pima	667.20	0.66	667.20	0.66	623.20	0.65	667.20	0.66	664.80	0.66
Soybean	251.00	0.91	249.40	0.91	218.20	0.91	250.40	0.91	248.60	0.91
Voting R.	354.40	0.93	183.60	0.89	197.00	0.93	116.40	0.89	286.60	0.93
Zoo	81.80	0.95	81.80	0.95	31.40	0.92	81.80	0.95	51.40	0.95

minimizing its size. In addition, the measures of case density and group density for modeling competent case–based reasoning systems are mentioned in further work of Smyth and McKenna [12].

Several additional quality measures that are based on various properties of the individual cases and of the case base as a whole are defined in the work by Reinartz, Iglezakis, and Roth–Berghofer [8]. The quality measures describe specific characteristics of the case base such as correctness, consistency, uniqueness, minimality, and incoherence. One noteworthy application of these quality measures is their use in triggering maintenance operations. Also, the first basic experimental results using case and case base properties to trigger maintenance were presented in [6].

The field of maintenance in case–based reasoning systems is still open for improvements and further work. As presented in the previous sections there are possibilities for maintenance using case properties. Some of these have proven to be very effective. The results presented above showed that the case base can be optimized quite effectively using one case property at a time. By implementing optimization using a combination of properties, the results could be improved. There is also the possibility that other case properties could be discovered and used to further improve the accuracy and size of case bases. A method to analyze correlations between case properties would be useful. In the above experiments cases have simply been added by means of a restore operation. The use of particular restore operations such as specializing cases or altering cases (cf. [9]) and the addition of expert knowledge could lead to a greater advancement in the applicability of these methods for optimization.

6 Acknowledgments

The authors are grateful to Thomas Reinartz (DaimlerChrysler, Research and Technology) for the fruitful discussions and continual support.

References

[1] David W. Aha. Case–based learning algorithms. In *Proceedings of the DARPA Case–Based Reasoning Workshop*, pages 147–158. Morgan Kaufmann, 1991.

[2] David W. Aha. A proposal for refining case libraries. In *German Workshop on Case–Based Reasoning (GWCBR)*, 1997.

[3] David W. Aha and Leonard A. Breslow. Learning to refine case libraries: Initial results. Technical Report AIC–97–003, Navy Center for Applied Research in AI, 1997.

[4] David W. Aha and Leonard A. Breslow. Refining conversational case libraries. In *Proceedings of the Second International Conference on Case–Based Reasoning*, pages 267–278, 1997.

[5] Catherine L. Blake and Christopher J. Merz. UCI repository of machine learning databases, 1998.

[6] Ioannis Iglezakis and Christina E. Anderson. Towards the use of case properties for maintaining case based reasoning systems. In *Proceedings of the Pacific Knowledge Acquisition Workshop (PKAW)*, pages 135–146, 2000.

[7] Ioannis Iglezakis and Thomas Roth–Berghofer. A survey regarding the central role of the case base for maintenance in case-based reasoning. In *Proceedings of the ECAI Workshop on Flexible Strategies for Maintaining Knowledge Containers*, pages 22–28, 2000.

[8] Thomas Reinartz, Ioannis Iglezakis, and Thomas Roth–Berghofer. On quality measures for case base maintenance. In *Proceedings of the 5th European Workshop on Case–Based Reasoning*, pages 247–259. Springer–Verlag, 2000.

[9] Thomas Reinartz, Ioannis Iglezakis, and Thomas Roth–Berghofer. Review and restore for case–base maintenance. *Computational Intelligence: special issue on maintaining CBR systems*, 17(2):214–234, 2001.

[10] Michael M. Richter. The knowledge contained in similarity measures. Invited Talk at the 1st International Conference on Case–Based Reasoning (ICCBR), 1995.

[11] Barry Smyth and Mark T. Keane. Remembering to forget: A competence–preserving deletion policy for case–based reasoning systems. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 377–382, 1995.

[12] Barry Smyth and Elizabeth McKenna. A portrait of case competence: Modelling the competence of case–based reasoning systems. In *Proceedings of the 4th European Workshop on Case–Based Reasoning.*, pages 208–220. Springer–Verlag, 1998.

[13] Jun Zhu and Qiang Yang. Remembering to add: Competence–preserving case addition policies for case base maintenance. In *Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI)*, pages 234–239, 1999.