Remembering to Add: Competence-preserving Case-Addition Policies for Case-Base Maintenance

Jim Zhu and Qiang Yang
School of Computing Science
Simon Fraser University
Burn a by, British Columbia
Canada, V5A 1S6

Abstract

Case-base maintenance is gaining increasing recognition in research and the practical applications of case-based reasoning (CBR). This intense interest is highlighted by Smyth and Keane's research on case deletion policies. In their work, Smyth and Keane advocated a case deletion policy, whereby the cases in a case base are classified and deleted based on their coverage potential and adaptation power. The algorithm was empirically shown to improve the competence of a CBR system and outperform a number of previous deletion-based strategies.

In this paper, we present a different case-base maintenance policy that is based on case addition rather than deletion. The advantage of our algorithm is that we can place a lower bound on the competence of the resulting case base; we demonstrate that the coverage of the computed case base cannot be worse than the optimal case ba.se in coverage[4] by a fixed lower bound, and the coverage is often much closer to optimum. We also show that the Smyth and Keane's deletion based policy cannot guarantee any such lower bound. Our result highlights the importance of finding the right ca.se-ba.se maintenance algorithm in order to guarantee the best case-base coverage. We demonstrate the effectiveness of our algorithm through an experiment in case-based planning.

1 Introduction

Case-ba.se maintenance refers to the task of adding, deleting and updating cases, indexes and other knowledge in a case base in order to guarantee the ongoing performance of a CBR system. Case-base maintenance is particularly important when a case based reasoning system becomes a critical problem solving system for an organization. This is because for any such organization, the knowledge may change over time and the need for different knowledge structures for problem solving may vary. The case-base size will increase with time, creating significant barrier to reasoning efficiency and the user's ability to understand the results.

In response to these problems, there has been a significant increase in case-base maintenance research. One branch of research has focused on the ongoing maintenance of case-base indexes through training and case base usage [Cunningham e*t* a/., 1997; Fox and Leake, 1995; Aha and Breslow, 1997; Zhang and Yang, 1998]. Another branch of research have focused on increasing the overall competence of the case base through case delet$_{ion}$ Smyth and Keane, 1995; Markovicli and Scott, 1988; Domingos, 1995; Aha *ct* a/., 1991; Smyth and Keane, 1995; Racine and Yang, 1997] in a way similar to utility-based control-rule deletion policies [Minton, 1990]. Excellent surveys of this field can be found in [Leake and Wilson, 1998] and [Watson, 1997].

This recent surge of interest in case-base maintenance is highlighted by Smyth and Keane seminal work on competence-preserving case-deletion policy [Smyth and Keane, 1995]. In this work, the cases in a case base are classified into a type hierarchy based on their coverage potential and adaptation power. The deletion policy then selectively deletes cases from a case base guided by the classification of the cases until a limit on the case base size is reached. The algorithm was empirically shown to preserve the competency of a CBR system and to outperform a number of previous deletion based strategies.

In this paper, we present a detailed analysis of Smyth and Keane's deletion based policy and show that, this policy does not always guarantee the competence preserving property. In particular, we show that using this policy can potentially result in a case base with significantly decreased performance. In response, we develop a different case-base maintenance policy that is based on case addition rather than deletion. By this policy, cases in an original case base are repeatedly selected and *added* to an empty case base until a certain size limit is reached, producing an updated case base which high coverage guarantee. The addition based policy will allow a more global view of the case base as a result of the maintenance operations. We show that both the Smyth and Keane's deletion-based policies and our addition-based policies have the same time complexity. The advantage

of the addition-based policy is that we can place a lower bound on the competence of the resulting case base while the deletion-based policy cannot; we demonstrate that the coverage of the computed case base cannot be worse than the optimal case base in coverage by a fixed lower bound, and often is much closer to the optimal coverage.

Our result highlights the importance of finding the right similarity metrics in order to guarantee the best case-base coverage. We contend that it is important to tie the definition of similarity-based metrics to adaptation costs. Based on this observation, we demonstrate through case-based planning how to construct high-quality similarity metrics that lead to highly competent case bases, and discuss various implications of our result in practical implementation of case-base maintenance systems. Finally, we confirm our competence-preserving claims through an experiment in case based planning.

2 Case-Base Maintenance and Case-deletion Policies

2.1 Related Work

Recently, there has been intense interest in case based reasoning research community on the problem of case-base maintenance. Leake and Wilson gave an in-depth summary and analysis of this field (Leake and Wilson, 1998]. For our purpose, the problem of case-base maintenance is divided into two broad categories: maintaining the case base indexes and maintaining the case base contents. In case-base index maintenance, [Cunningham *el* a/., 1997) presents an introspective learning approach to learn adjusted case base indexes by monitoring the runtime processes of a case based reasoner. An extended approach is developed in [Zhang and Yang, 1998], where a layered architecture is adopted for representing case base indexes and a neural network algorithm is adapted for maintaining the feature weights. Fox and Leake [Fox and Leake, 1995] and Aha and Rreslow [Aha and Breslow, 1997] consider case-base index-revision policies that improve the performance of a case base in response to events such as plan failures.

Researchers in case-base content maintenance are mainly concerned with the issue *optimization.* Due to the large size of some case bases, it is necessary to delete cases as time goes by, and when retrieval become increasingly expensive [Smyth and Keane, 1995]. This issue is called the *swamping problem.* The main strategy is that of deciding which cases to delete based on an adaptation structure. These strategies include one for random deletion as advocated by [Markovich and Scott, 1988], and more sophisticated deletion based on the frequency with which each case is retrieved and deleted if they are not frequently accessed [Minton, 1990]. The problem with both of these approaches is that "important" cases can be deleted by mistake. Various approaches have been designed to address this problem. Domingos [Domingos, 1995] and Aha, Kibler and Albert [Aha *et* a/., 1991] consider instance-based learning approaches for generalizing to reduce the size of a case base without decreasing its problem-solving power. Smyth and Keane [Smyth and Keane, 1995] consider a competence-preserving approach to case deletion. Watson [Watson, 1997] presents methodologies for a human designer of a case base to consider for case-base maintenance. Racine and Yang [Racine and Yang, 1996] consider the problem of removing redundancy and inconsistency from a large semi-structured case base in order to improve the case base performance.

2.2 Coverage and Neighborhood Functions

We define a case as a problem-solution pair. That is, each element *C* of a case base is a pair *C* = (X, .s), where $s \in S$ is a corresponding solution to a problem description *x.* For each problem x_1 in a case base X_1, x_1 can represent the case $(x_1, \pi(x_1))$.

Hence we also call x_1 a case. Let $N(x_1)$ be the set of cases x_2 whose solution $\pi(x_2)$ is close to $\pi(x_1)$. More formally,

$$N(x_1) = \{x_2 \mid D(\pi(x_1), \pi(x_2)) \leq L\}$$

where *L* is a constant limit on the cost of adapting a solution. Essentially, *N* defines a coverage of *x\.* We call *N(x\)* the coverage or neighborhood of *x\.* Later in the paper (Section 6), we define a distance metric $d(x_1<x_2))$ for case based planning using the number of adaptation steps to apply to the solution of x_1 in order to solve x_2. For now, we assume that the neighborhood function is given as done by Smyth and Keane [Smyth and Keane, 1995], and consider how to use this information to compute a near-optimal case base A'i from a given case base A'.

Based on the above notion, the coverage of a case is determined by a similarity metric and adaptation costs. We can consider the coverage of a case as the neighborhood of the case within certain adaptation limits. Hence, we consider the notion of a problem neighborhood and coverage interchangeable. Similarity metrics are used to measure the similarities between cases. They are usually numerical, but a good similarity metric is not easy to find in many application domains. We therefore introduce the notion of a *neighborhood* which is a more intuitive notion.

2.3 Analyzing Case Deletion Policy

When the size of a case base gets large, there[1] is a need to select a subset of the cases to keep. To address this problem, Smyth and Keane [Smyth and Keane, 1995] suggest a case deletion based approach. The premise of this approach is that each case in the case base should be classified according to its competence. These classifications are made according to two key concepts: *coverage* and *reachability.* Coverage refers to the set of problems that each case can solve. Reachability is the set of cases that can be used to provide solutions for a problem.

Cases that represent unique ways to answer a specific query are *pivotal cases. Auxiliary* cases are those which are completely subsumed by other cases in the base. In

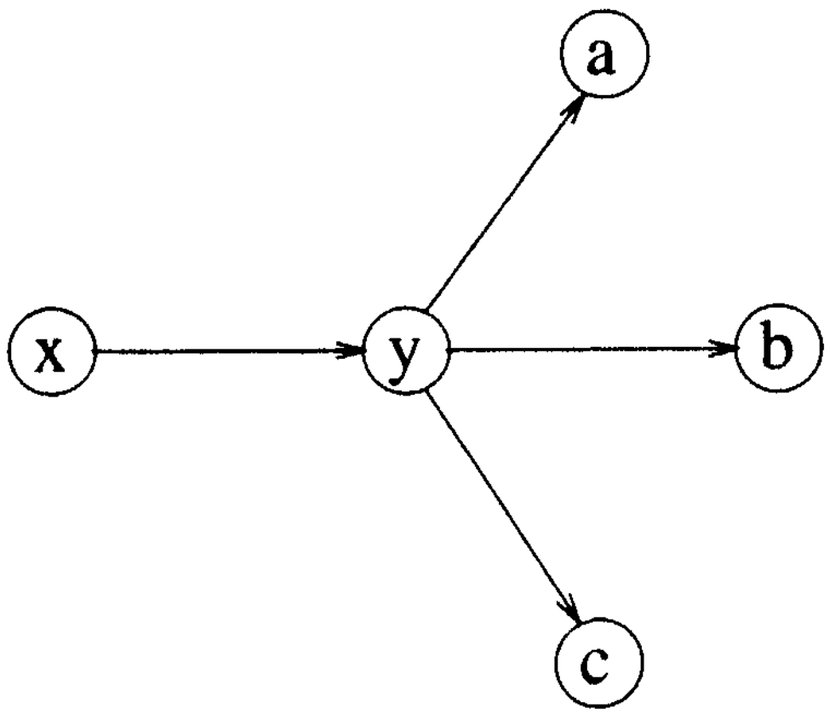

Figure 1: Case Base Structure Graph

between these two extremes are the *spanning* cases which link together areas covered by other cases, and *support* cases which exist in groups to support an idea. The deletion algorithm deletes cases in the order of their classifications : auxiliary, support, spanning and then pivotal cases [Smyth and Keane, 1995],

Similarly, the definitions for spanning and support cases also rely on the concepts of coverage and reachability. In their evaluation of their algorithm, which consisted of 50 cases, Smyth and Keane restrict the size of the case base and the size of the problem space and manually identify the category in which each case falls.

The deletion based policy is motivated by the need to delete cases in order to maintain the competency of a case base at a reasonable size. However, no mention is made about why auxiliary cases should be deleted first, and how the quality of the resulting case base is ensured after the update is done. Pivotal cases may be "important" or they may simply contain anomalies that distinguish them from the rest of the case base. But deleting the rest of the case base first only offers an intuitive solution to the case-base maintenance problem; there was no guarantee on the level of true competence preservation.

Smyth and Keane's terminologies can be illustrated graphically as in Figure 1. In this figure, an arrow from a node *x* to a node *y* means that the case *y* is in the neighborhood of the case *x.* Therefore, in Figure 1, $N(x) = \{x, y\}$, $N(y) = \{y, a, b, c\}$, $N(a) = \{a\}$, *N(b)* = {6} and *N(c)* = {c}. According to Smyth and Keane's classification scheme, *x* is pivotal, *y* is spanning and a, 6, *c* are auxiliary.

From these definitions, it is easy to see that pivotal problems are the most unique, spanning problems are less unique, but auxiliary problems are not unique. Smyth and Keane's footprint deletion (FD) and footprint-utility deletion (FUD) policy delete auxiliary problems first, then support problems, then spanning problems, finally pivotal problems. This approach is better than a random deletion policy for preserving competence. The competence of a case base built by Smyth and Keane's footprint deletion (FD) or footprint-utility deletion (FUD) policy is not guaranteed to be preserved.

Theorem 1 *FD and FUD can lose almost all the competence in the worst case.*

Proof:

To prove this theorem, we only need to give an example. Suppose that in some domain, we have the graph structure as shown in Figure 1.

In this figure, each node stands for a problem (or case). We see that the problem *x* is pivotal, *y* is spanning and a.,6,c are auxiliary. Suppose that we want to build a case base with only one element; that is, we restrict our case base to be of size one. By the footprint deletion or footprint-utility deletion policies, cases *a,b,c* should be deleted first, followed by *y.* As a result, only problem *x* is left in the case base. The coverage is $\frac{2}{5}$ of the original competence. If we increase the number of auxiliary cases (such as a, b, c in the figure) to *k,* then the coverage is $\frac{2}{k+2}$ of the original competence. The percentage approaches to zero as $k \longrightarrow \infty$. Therefore, the quality of the case base is arbitrarily bad. This completes the proof. D

3 Case-addition Based Policy

Suppose that the neighborhoods of all cases in a case base are obtained. We say a case is *good* if its neighborhood is large. To select good cases, the distribution of the cases or the frequency of cases occurring should also be considered. For instance, in an example travel domain (Section 6.2), suppose that more people prefer a travel plan between City 1 and City 4, then we should put this plan in a case base in order to minimize the cost of searching for such plans. Taking this into account, we define case coverage as follows:

Given a domain, we have a case space A" and a solution space *Y.* Let $x \in X$ be a case. Denote *N(x)* the neighborhood of *x* and $N(X_1) = \cup_{x \in X_1} N(x)$., *N(Xi)* contains cases which are close to some other cases in X_1. Suppose that *P* is a a frequency function of the cases (between 0 and 100%); equivalently there is a distribution of cases. Then the case base coverage of X_1 is defined as

$$Coverage(X_1) = \frac{\sum_{x \in N(X_1)} P(x)}{\sum_{x \in X} P(x)}$$

Since X_1 is a subset of A', the case coverage is a real number between 0 and 1. If the function *P* does not exist, we assume that *P* = 1 - the constant function.

The benefit of a case *x* with respect to a set W of cases is defined as Bft(x) = $\sum_{y \in N(x) - N(W)} P(y)$, where $N(W) = \cup_{w \in W} N(w)$., The benefit of a case set $\{x_1, x_2, \bullet\bullet\bullet, X_k\}$ is defined as $\sum_{y \in \cup_{i=1}^{k} N(x_i) - N(W)} P(y)$-

Suppose we want to build the case base *X* with at most *k* cases based on a set *Z* of cases. We formulate this optimization problem as follows

(*) Choose cases $\{x_1, x_2, \cdots, x_k\}$ from case set *Z* to maximize the benefit of $\{x_1, x_2, \cdots, x_k\}$ $(W = \emptyset)$.

This optimization problem is NP-compete. One can easily prove this by a reduction from Set-Covering [Garey and Johnson, 1979]. Thus, we look for heuristics to find approximate solutions. Consider the following case-addition algorithm based on selecting cases from *Z* and adding them to the new case base:

Case-Addition Algorithm:

1. Determine the neighborhood *N(x)* for every case $x \in Z$.
2. Set $X_1 = \emptyset$.
3. Select a case from *Z* — *X* with the maximal benefit with respect to *N{X\)* and add it to *X*.
4. Repeat step 3 until *N(Z)* - *N(Xi)* is empty or X_1 has *k* elements.

Remark. Here we consider the case coverage as the benefit. In fact, the benefit can be defined on other notions as long as it captures the concept of usefulness.

The Case-addition Algorithm is a greedy algorithm. Therefore it may not give the best choice of X_1 with respect to the case coverage. However, we can prove that its case coverage is at least 63% of the optimal case base, for any fixed case-base size *k.*

Theorem 2 *The case addition algorithm produces a case base X_1 such that the coverage of X_1 is no less than* 63% *of the coverage of an optimal case base.*

The proof for this theorem is derived from that of a greedy algorithm for set covering. Due to space limitations, we only provide an intuitive sketch for the proof here. Suppose that $N(Z) - N(X_1) \neq \emptyset$, then *X* = {$x_1$, X_2, ■ ■ •, *Xk)* has *k* elements and labeled by the order of selection. Let a; be the benefit of $x_i, 1 \leq i \leq k$ Suppose that {y_1, y_2, ..., y_k} is an optimal choice for X_1. Let b_i, $1 \leq i \leq k$, be the benefit of y_i; under the index order. Note that $a_1 \geq a_2, \cdots, \geq a_k$. We can then relate a_i with *bj* and derive the following inequality: the ration of the coverage of $\{x_1, x_2, \cdots, x_k\}$ and the optimal case base coverage for *k* cases is

$$\frac{\sum_{i=1}^{k} a_i}{k a_1} \geq \frac{1}{k} \sum_{i=1}^{k} \frac{a_i}{a_1} = 1 - \left(\frac{k-1}{k}\right)^k$$

Our result is motivated by a similar result in [Harinarayan *et* a/., 1996] for constructing a data cube used in building a data warehouse. The main difference is that in CBR case adaptation is our main concern whereas in data warehousing the utility of a data view is of great importance.

Analyses of both Smyth and Keane's case-deletion algorithm and our case-addition algorithms reveal that they require the same time complexity $(O(n^2))$. This is because both algorithms are dominated by computing the coverage of cases, hence their costs should be about the same. When n is large, computing the competence

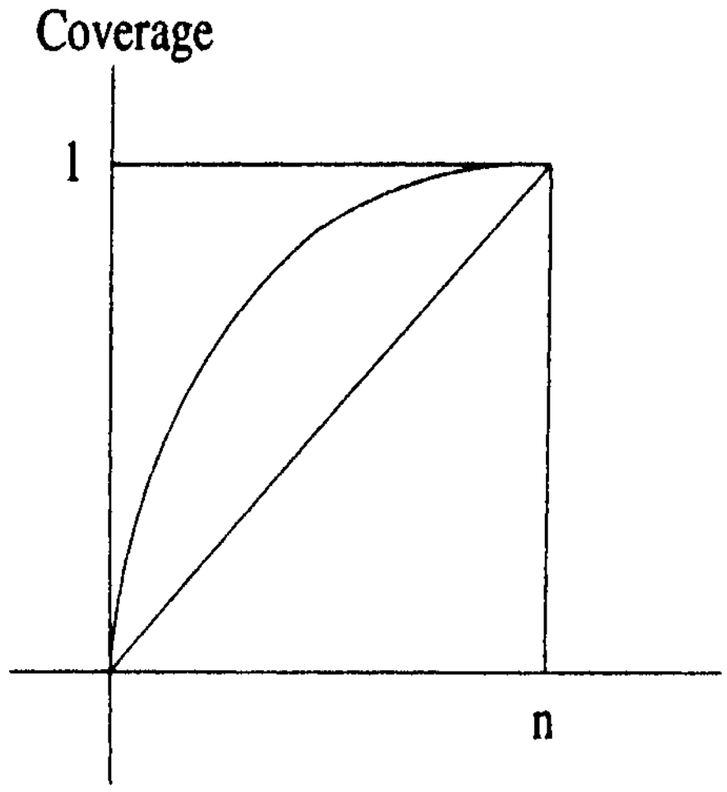

Figure 2: Graph of Coverage

categories and the coverage of cases becomes expensive. However, the key point is it is only computed once as a start-up cost. We have developed a practical incremental case-addition algorithm which will will present in an extension of the paper.

4 Relating Case Base Size with Coverage

We have so far assumed that the case base size *k* is given. Having a different *k* will result in a case base with a different coverage value. How should a case base maintainer choose an appropriate size for a case base?

In this section we will estimate how a case base size determines its coverage. In our notation, we would like to estimate the ratio between $|X_1|$ and $|N(X_1)|$. Suppose that x_1, x_2, •.... $^x n$ are cases selected by case-addition algorithm with the benefits of *a*, a_2, • • •, a_n. The$_n$

$$a_1 \geq a_2 \geq \cdots \geq a_n.$$

Hence, the ratio $|X_1|/|N(X_1)|$ is between $\frac{1}{a_1}$ and $\frac{1}{a_n}$ However, this estimation is rather rough.

Our theorem below points out the precise relationship between case-base sizes and case-base coverage:

Theorem 3 *Let M be the size of the original case base before applying the case-addition algorithm for maintenance. Let k be the size of the case base after maintenance. Suppose that all cases have equal probability of appearance; that is, the frequency of all cases are the same. Suppose also that when constructing the result case base the benefits of new cases decrease linearly. Then we have*

$$\text{Coverage} \geq R(2 - R)$$

where R = k/M.

Again due to space limitations we omit the proof. Instead, we plot the coverage curve as shown in Figure 2.

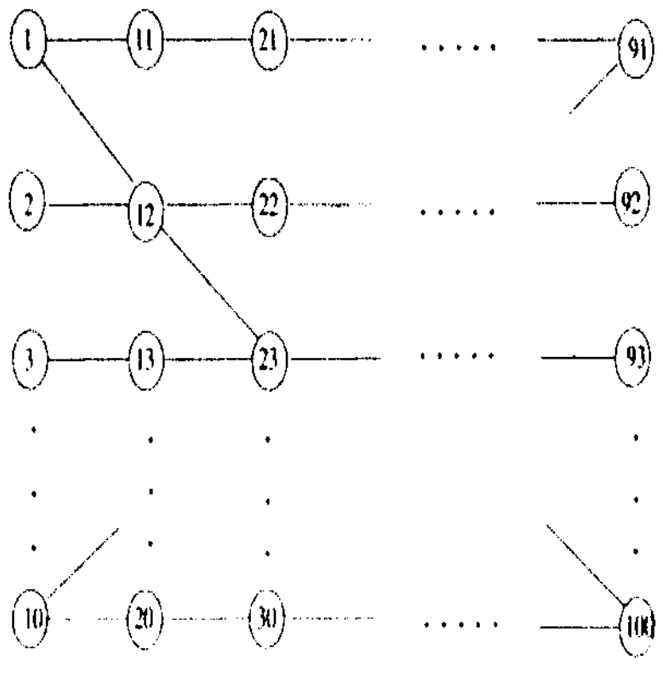

Figure 3: State Graph of a Travel Domain

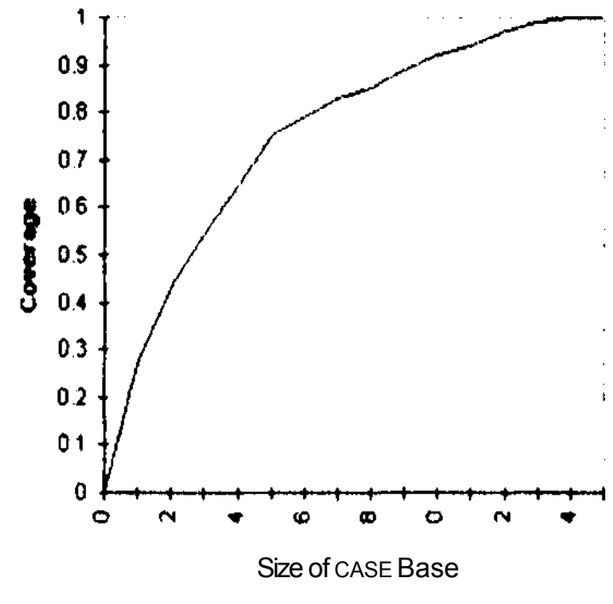

Figure 4: Coverage Graph of a Travel Domain

From the graph, we see that over 15% of cases can attain about 50% coverage while 50% of cases can attain about 85% coverage. In general, things are better. We have used the linear approximation to draw the coverage graph. In general, the difference $(a_k + 1 - a_k)$ decreases quickly as k increases. Hence, we can have a better model via replacing $\sum_{i=0}^{k}(a_1 - \bar{a}i)$ with $\sum_{i=0}^{k}(a_1 - \bar{a}i^{1/h})$, where h is an integer > 1. The resulting graph of coverage is similar, but the desired coverage can be reached much sooner.

5 Experiments with the Case-Addition Algorithms

To fully validate our case-addition algorithm, we need to perform empirical tests in various domains. In the remaining of this paper, we present one such experiment in a travel planning domain. In this domain, a problem x can be considered a pair of states: $x = (s_i, s_y)$, where s_t is an initial state and s_g a goal state. A case is a pair (x,p), where x is a problem definition and p a plan. We consider a plan as a sequence of actions taking one from the initial state to the goal state. There are many ways to define planning algorithms for finding a plan [Kambhampati *et al,* 1995; Yang, 1997]; here we focus on how to reuse the previous plans.

In planning various similarity metrics have been proposed for determining the distances between two plans Hammond, 1990; Hanks and Weld, 1995]. For example, the foot-printed similarity metric [Veloso, 1992; 1994] compares relevant features of a case with features of a new problem where a relevant feature is considered relevant to a goal and a solution if the feature contributes to achieving the goal in the solution. rao:cbp97 presents a mechanism for adding plans to a plan library in order to limit the size of the case base. The adaptation-guided metric [Kinley *ct al,* 1995] exploits the adaptation knowledge of a case base.

In our example travel domain, as shown in Figure 3, there is a map in which an agent will travel between two cities. The agent would like to remember a few useful paths so that a case base can be constructed. In order to do this, it is assumed that a neighborhood function is defined based on adaptation as follows: Let a case x be a problem together with a solution *soln,* where $x = (s_i, s_g)$ and *soln* - $s_1 \longrightarrow s_2 \longrightarrow \cdots \longrightarrow s_k$. Here $s\backslash$ = -s, and $S_k = s_g$. Then the neighborhood $N(x)$ of problem x is defined as the neighborhood of the solution. More precisely,

$$N(x) = (\cup_{j=1}^{k} N(s_j)) \times (\cup_{j=1}^{k} N(s_j)) \qquad \text{(i)}$$

where the neighborhood of a state $N(S_J)$ is defined as the set of states (that is, cities) that are one step from S_j. We call this neighborhood function the "state-based" similarity measurement for planning.

In our experiment, an input problem is defined as any pair of cities, and the solution, which is a case, is a path going from an initial city to a destination city.

The state can be moved horizontally or along the main diagonal lines. The number of states = 100. The problem space has 100" = 10000 problems. Randomly select 80 problems from the problem space and solve these problems from scratch by a forward chaining and breath-first planner. By computing the neighbors of these problems, we see that the union of all these neighbors covers 3138 problems. Applying the case-addition algorithm, we get the result which is shown in Figure 4.

6 Conclusions and Discussion

We conclude that it is important: to tie similarity metrics with adaptation costs. Based on this concept, we have given an approximation algorithm for building a case base with near-optimal property. We attribute this property to the fact, that we use a case-addition rather than a case-deletion policy as done by Smyth and Keane. In the future, we will study how to maintain a case base in order to increase the quality of cases in addition to increasing the case bsae coverage.

Acknowledgment

The authors are supported by grants from: Natural Sciences and Engineering Research Council of Canada (NSERC), an Ebco/Epic NSERC Industrial Chair Fund,

BC Advanced Systems Institute and Canadian Cable Labs Fund. We wish to thank IJCAT reviewers for their in-depth suggestions.

References

[Aha and Breslow, 1997] D.W. Aha and L. Breslow. Refining conversational case libraries. In *Proceedings of the Second International Conference on Case-Based Reasoning, ICCBR-97,* pages 267-276, Provedence RI, USA, 1997.

[Aha *et* a/., 1991] D. Aha, D. Kibler, and M. Albert, Instance-based learning algorithms. *Machine Learning,* 6:37 66, 1991.

[Cunningham *et ai,* 1997] P Cunningham, A. Bonzano, and B. Smyth. Using introspective learning to improve retrieval in car: A case study in air traffic control. In *Proceedings of the Second International Conference on Case-Based Reasoning, ICCBR-97,* pages 291 302, Providence RI, USA, 1997.

Dorningos, 1995] P. Domingos. Rule induction and instance-based learning. In *Proceedings of the Thriteenth International Joint Conference on Artificial Intelligence,* pages 1226 1232, San Francisco, CA, USA, 1995. Morgan Kaufmann.

[Fox and Leake, 1995] S. Fox and D. B. Leake. Learning to refine indexing by introspective reasoning. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence,* Montreal, Canada, August 1995.

[Garey and Johnson, 1979] Michael R. Carey and David S. Johnson. *Computers and intractability, A Guide to the Theory of NP-Completeness.* W. H. FREEMAN and COMPANY, New York, 1979.

[Hammond. 1990] Kristian Hammond. Explaining and repairing plans that fail. *Artificial Intelligence,* 45(1): 173 228, 1990.

Hanks and Weld, 1995] Steve Hanks and Daniel S. Weld. A domain-independent algorithm for plan adpatation. *Journal of Artificial Intelligence Research,* 2:319 360, January 1995.

[Harinarayan *et al,* 1996] V. Harinamy an, A. Rajaranian, and J. D. Ullman. Implementing data cubes efficiently. *ACMSJGMOD,* pages 311 320, 1996.

[Kambhampati *et ai,* 1995] Subbarao Kambhampati, Craig A. Knoblock, and Qiang Yang. Planning as refinement, search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence,* 75(3), 1995. Special Issue on Planning and Scheduling, edited by J. Hendler and D. McDermott.

[Kinley *et ai,* 1995] A. Kiniey, D.B. Leake, and D. Wilson. Learning to improve case adaptation by introspective reasoning and cbr. In *Proceedings of the First International Conference on Case-Based Reasoning,* pages 229 240, Sesimbra, Portugal, 1995. Springer-Verlag.

[Leake and Wilson, 1998] D. B. Leake and D. C. Wilson. Categorizing case-base maintenance:dimensions and directions. In *Proceedings of the J 998 European Workshop on CBR (WECBR-98),* 1998.

[Markovich and Scott, 1988] S. Markovich and P. Scott. The role of forgetting in learning. *Proceedings of the Fifth International Conference on Machine Learning,* 1:459-465, 1988.

[Minion, 1990) S. Minton. Qualitative results concerning the utility of explanation-based learning. *Artificial Intelligence,* 42:363-391, 1990.

[Racine and Yang, 1996] Kirsti Racine and Qiang Yang. On the consistency management of large case base: the case for validation. *AAA1 Technical Report-Verification and Validation Workshop,* 1996.

[Racine and Yang, 1997] Kirsti Racine and Qiang Yang. Maintaining unstructured case bases. In *Proceedings of the Second International Conference on Case-Based Reasoning, ICCBR-91,* pages 553 564, Providence RI, USA, 1997.

[Smyth and Keane, 1995] B. Smyth and M. Keane. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *International Joint Conference on Artificial Intelligence,* volume 1, pages 377 382, 1995.

[Veloso, 1992] Manuela M. Veloso. *Learning by Analogical Reasoning in General Problem Solving.* PhD thesis, Carnegie Mellon University, Pittsburgh, USA, 1992.

[Veloso, 1994] Manuela M. Veloso. *Planning and Learning by Analogical Reasoning.* Springer Verlag, 1994.

[Watson, 1997] Ian Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems.* Morgan Kaufmann Publishers Inc., 1997.

[Yang. 1997] Qiang Yang. *Intelligent Planning A Decomposition and A bstracation Based A pproaeh.* Springer-Verlag, 1997. ISBN 3 540-61901-1.

[Zhang and Yang, 1998] Zhong Zhang and Qiang Yang. Towards lifetime maintenance of case based indexes for continual case based reasoning. In *Proceedings of the 8th International Conference on Artificial Intelligence: Methodology, Systems, applications,* Sozopol, Bulgaria, 1998.