Appears in Computational Intelligence (2001)

Computational Intelligence, Volume , Number ,

CASE-BASE MAINTENANCE BY INTEGRATING CASE INDEX REVISION AND CASE RETENTION POLICIES IN A DERIVATIONAL REPLAY FRAMEWORK

Héctor Muñoz-Avila

Department of Computer Science, University of Maryland, College Park, MD 20742-3255

This article presents case-base maintenance policies for case index revision and case retention. The policies are formulated in the context of case-based planners performing case adaptation by derivational replay. We implemented these policies on a particular case-based planner and claim that our case index revision policy improves the accuracy of the retrieval and that the case retention policy filters redundant cases better than other case retention policies known from the literature. Our claims are validated by empirical validation. We will observe that there is an inter-relation between the two policies that improves the filtering process of the case retention policy.

Key words: case-based maintenance, derivational analogy, feature weighting, planning

1. INTRODUCTION

Researchers in machine learning have long observed that uncontrolled growth of knowledge will reduce the overall *utility* of a learning system (Minton, 1990; Markovich & Scott, 1993) but only recently with the growing number of applications of case-based reasoning (Althoff et. al., 1995), researchers and practitioners have increasingly drawn their attention towards case-base maintenance (CBM). Studies include policies for case deletion (Smyth & Keane, 1995), over-all reduction of case libraries (Lei et. al., 1999) and algorithms for detection of case redundancy and inconsistency (Racine & Yang, 1997). Recently, a framework has been proposed for categorizing CBM systems (Leake & Wilson, 1998).

In this article we report a study on CBM policies for general-purpose case-based planners that perform case adaptation by Derivational Replay. In this adaptation method, an underlying first-principles planner is guided by following the sequence of planning decisions (the derivational trace) used to solve previous problems (Veloso & Carbonell, 1993). Adaptation by derivational replay is a powerful method and has been the subject of study with a variety of planning paradigms including non-linear planning (Veloso, 1994), partial-order planning (Ihrig & Kambhampati, 1997) and hierarchical planning (Muñoz-Avila et. al., 1994).

Our CBM policies are based both on the *outcome* and the *benefits* of the retrieved cases during case-based problem solving episodes. Intuitively, the outcome of adapting the retrieved case(s) is considered successful if the case(s) can be extended to a solution without revising the planning decisions prescribed by the case(s). In this paper we will refer to those cases as *extensible* cases. Otherwise it is considered a failure (Ihrig & Kambhampati, 1997). The benefit of the retrieved case(s) is a measure of the adaptation effort (Muñoz-Avila, 1999). If the adaptation effort is low, the retrieval is considered beneficial. Otherwise it is considered detrimental. As we will show later, the outcome and the benefit of the retrieved cases have no logical relation (i.e., neither implies the other one). However, we will use these concepts to state CBM policies and show an inter-relation between these policies. The following are the contributions of this article:

- We present a case index revision policy based on the outcome of the retrieval.
- We state a case retention policy based on the benefit of the retrieval.
- We perform an empirical evaluation of these policies and show that the case index revision policy improves the accuracy of the retrieval. We also show that the case retention policy is a better filter of redundant cases than other case retention policies known from the

literature for general purpose case-based planning.
- We show that the case index revision policy improves the filtering process of the case retention policy.
- We classify previous work on general purpose case-based planning using Leake and Wilson's CBM framework.

This article continues as follows: in the next section we motivate our study of CBM for general purpose case-based planners. Section 3 discusses derivational replay. Next, we present the CBM policies developed in our work and discuss their relation. Then, in Section 5 we present an empirical evaluation. After that, other CBM approaches to general-purpose case-based planning are compared using Leake and Wilson' framework. Finally, some concluding remarks are made.

2. MOTIVATION

The study of CBM policy in the context of synthesis tasks such as planning is not new. Fox and Leake (1995), for example, introduced a policy for revising indexes during case based problem solving episodes. Basically, if a more suitable case existed in the case base than the one initially selected during the retrieval process, an introspective analysis of the retrieval is made to avoid making the same mistake in future retrieval episodes. This approach requires, in addition to the planning knowledge, an introspective model of the retrieval method. A motivation of our work is to develop CBM policies that remove the need for such additional knowledge.

Another well-known CBM policy is based on the notions of *coverage* and *reachability* sets (Smyth & Keane, 1995; Smyth & McKenna, 1999). In the context of case-based planning we can re-phrase these concepts as follows: given a plan pl, its coverage set, written *CoverageSet(pl)*, is formed by all problems $\{p\}$ such that pl can be adapted to solve p. Given a problem p its reachability set, written *ReachabilitySet(p)*, is formed by all plans $\{pl\}$ such that pl can be adapted to solve p.

One of the first questions that may be asked for any general purpose planning algorithm is whether it is complete or not. A planning algorithm is *complete* if given any solvable problem, the algorithm can produce a plan solving that problem. Several planning algorithms such as SNLP (Barret & Weld, 1994) have been shown to be complete. In the context of case-based planning, Hanks and Weld (1995) demonstrated the completeness of an adaptation algorithm based on SNLP. Muñoz-Avila (1998) showed the completeness of adaptation by derivational replay based on SNLP and conjecture that if its underlying first-principles planning system is complete, adaptation by derivational replay will also be complete.

If an adaptation algorithm is complete, then given any solvable problem p and any plan pl, then the following holds:

- $ReachabilitySet(p) = \{pl : pl\ is\ a\ plan\}$
- $CoverageSet(pl) = \{p : p\ is\ a\ solvable\ problem\}$

This follows directly from the notion of completeness. If p is a solvable problem, then given any plan pl, pl can be adapted to solve p. Otherwise, the algorithm is not complete. Notice that this doesn't imply that using a single plan for adaptation is adequate as the effort required to adapt it might be prohibitive large, even larger than planning by first principles (Nebel & Koehler, 1995).

The above considerations show that the notions of coverage and reachability sets are not particularly useful in the context of a case-based planner built on top of a first-principles planner as they do not allow to group cases in several classes making them indistinguishable. Part of the difficulty of applying this and other techniques in this context is the role of the cases. In general purpose case-based planning, cases are used to guide the search (Veloso & Carbonell, 1993) as opposed to other approaches in which cases enhance the knowledge about the domain itself (Hammond, 1986). This of course doesn't mean that the work on reachability and coverage or approaches requiring additional knowledge besides planning knowledge are not useful for synthesis tasks such as planning. It is well known that first-principles planners require a symbolic representation of the knowledge, something that is infeasible for many domains. For those domains, however, in which such a symbolic representation can be made (e.g., (Koehler & Shuster, 2000)), we want to introduce a suitable CBM approach.

3. ADAPTATION BY DERIVATIONAL REPLAY

In adaptation by derivational replay, cases contain the derivational trace that led to a plan instead of the plan itself (Veloso, 1994). A derivational trace is a sequence of planning decisions that led a first-principles planning system to create a plan. Figure 1(a) depicts a typical planning decision that a first-principles planner must confront: given a goal G, the planner must choose between several operations (i.e., *opn-1*, *opn-2*,...,*opn-m*). Depending on which operation it chooses, a new set of subgoals $G1$, $G2$, ..., Gn is obtained. Some choices may lead to a solution, others might lead to a dead end requiring costly backtracking to explore alternative decisions . Furthermore, in some situations none of the alternatives may lead to a solution if in a previous planning decision a wrong choice was made.

Derivational replay guides the planning decisions. Figure 1(b) illustrates the replay step for the planning decision depicted in Figure 1(a). The derivational trace of a case indicates that to achieve a goal G' the operation *opn-i'* was chosen, which resulted in a new set of subgoals $G1'$, $G2'$, ..., Gn'.

Derivational replay. Derivational replay of a planning decision is performed in three steps (see Figure 1(b)):

1. Unifying G with G'.
2. If this succeeds, a search is made among the operations achieving G (i.e., *opn-1*, ...,*opn-m*) for one which unifies with *opn-i'*.
3. If such operation, *opn-i*, exists, it is applied to decompose G into subgoals $G1$, $G2$, ..., Gn, which are unified against the subgoals of the case $G1'$, $G2'$, ..., Gn'.

How the adaptation process continues depends on the particular implementation of adaptation by derivational replay. Some implementations will continue the process by repeating the same three steps recursively with each subgoal Gi that unifies with a subgoal Gi' in the derivational trace until either all subgoals are solved or no subgoals in the derivational trace can be found that unify with the unsolved subgoals of the current problem. In the latter case, first-principles planning can be used to solve the remaining subgoals (e.g., Ihrig & Kambhampati, 1997). Other implementations may use heuristic knowledge to interrupt the replay process and select between alternative cases and first-principles planning to solve the remaining subgoals (Veloso & Carbonell, 1993). However, independent of the particular implementation, adaptation by derivational replay always involves the combination of first-principles planning and derivational replay. That is, a derived plan will typically con-

tain parts which were obtained by derivational replay and parts which were obtained by first-principles planning.

It is well known that the benefit of adaptating a prior case depends on large measure on the accuracy of the retrieval (Smyth & Keane, 1994). An inadequate retrieval procedure may lead to plans that are difficult or not adaptable at all. For the particular situation of adaptation by derivational replay, though cases can always be adapted, an inadequate retrieval procedure may result in a significant search effort by the underlying first-principles planner. In a worst case scenario the first-principles planner may need to backtrack on all decisions made during replay. Thus, the resulting effort might be larger than if the first-principles planner would have solved the problem from scratch even without taking into account the overhead caused by the retrieval process. Thus, an accurate retrieval technique combined with complementing CBM techniques that target reduction in case redundancy and increase in case competence is key for the effectiveness of adaptation by derivational replay.

4. TWO CBM POLICIES FOR DERIVATIONAL REPLY

In this section we present two policies for CBM in the context of case-based planners adapting cases by derivational replay: specifically, a case index policy and a case retention policy. We will describe each separately and then we will discuss how these policies are related.

4.1. Case Index Refinement Policy

Our case index refinement policy follows a punishment/reward system based on the outcome of the adaptation process. The outcome of the retrieval is considered a success if planning decisions made by replaying the derivational trace of the retrieved case can be extended to solve the problem. In this situation, we say that the case is extensible. Otherwise, some decisions replayed from the case need to be revised by the underlying first-principles planner and a retrieval failure occurs (Ihrig & Kambhampati, 1997). Figure 2 illustrates a non extensible case. In a typical situation, part of the retrieved case (labeled B) is not replayed in the new situation. This happens when the operation (corresponding to B) in the derivational trace is not applicable in the current problem and as such it does not unify any of the operations solving the current goal (step 2 of the derivational replay process discussed in the previous section). When completing the solution, the first-principles planner is forced to revise a planning decision replayed from the case (labeled A).

Case Index Revision Process. Ihrig & Kambhampati's (1997) retention policy retains non extensible cases. In our approach depending if a case is extensible or not different case index revisions are made. Cases consist of a problem and the derivational trace that lead to a plan solving that problem. A *problem* consists of a collection of features and goals. Features are statements about the world known to be true and goals are the statements that have to be achieved. Features and goal statements are represented as ground predicates. Each feature, i, has an associated weight, $\omega_{i,C}$, depending on the particular case C in which that feature occurs. When a new case is created the associated weight for each feature is set to one. With time these weight changes predict the relative importance of the feature in the case. Given a candidate case C and a problem P, our similarity assessment, $sim_{wg}(C, P)$, counts the weighted proportion of features in the case that occur in the new problem:

$$sim_{wg}(C,P) = \begin{cases} \sum_{i \text{ common to } P \text{ and } C} \omega_{i,C} & : \; if\ G_C \subset G_P \\ 0 & : \; otherwise \end{cases}$$

where G_C and G_P are the goals of the C and P respectively and $\subset$ is the subset-set relation.

The purpose of our case index revision policy is to tune the feature weights according to relative importance of a feature in a particular case. The relative importance of a feature relative to a case can be stated as follows:

> If the weights of the features, i_1, i_2, ..., i_n of a case C were normalized so that $\omega_{i_1,C} + \omega_{i_2,C} + ... + \omega_{i_n,C} = 1$, then the factor $\sum_{j \neq k} \omega_{i_j,C} / \sum_j \omega_{i_j,C}$ expresses the *reliability* of making an adequate retrieval when the feature i_k is the only one not occurring in the new problem.

The features and goals serve as indexes for the cases during the retrieval process. At each CBR problem solving episode, the feature weights of the retrieved case(s) may be revised. The case index revision process for each retrieved case C occurs in three steps:

1. The outcome of the retrieval is stated for C (i.e., if the retrieved cases are extensible or not).
2. The set, $feat_C$ of all features in C that didn't occur in the new problem is determined.
3. The weight of each feature in $feat_C$ is revised.

For revising the feature weights, a feedback model based on incremental optimizers is used (Salzberg, 1991). Each case, C, maintains two counters: k^C and f^C. The first one indicates the number of times the case C was extensible and the second one the number of times it was non extensible. The weight, $\omega_{i,C}$, of a feature i is updated according to the following equations:

$$\omega_{i,C} = \begin{cases} \omega_{i,C} - \triangle_{k^C,f^C} & : \; if\ the\ case\ is\ extensible \\ \omega_{i,C} + \triangle_{k^C,f^C} & : \; otherwise \end{cases}$$

where the factor $\triangle_{k^C,f^C}$ meets the constraint: $0 \leq \triangle_{k^C,f^C} \leq \beta \times n^C$. The number of features in the initial state of C is denoted by n^C. Thus, the change in the weight of the features is bound by a factor, $\beta \times n^C$, directly proportional to the number of features in the case ($\beta \geq 1$). If the value of $\omega_{i,C}$ is smaller than 1, then $\omega_{i,C}$ is assigned the value 1 and the weights of the other features in the case are incremented proportionally.

The rationale behind the revision is the following: if a case feature does not occur in the current problem but the case was extensible, the features weight is decreased because its absence didn't affected the retrieved case being extensible. Conversely, if the case is not extensible, its weight is increased because its absence from the current problem might have contributed to the case not being extensible. By revising the feature weights over a period of several CBR problem solving episodes, we expect that the likelihood of a case being non extensible decreases. The incremental factor $\triangle_{k^C,f^C}$ is computed as follows:

$$\triangle_{k^C,f^C} = \begin{cases} \beta \times n^C - (k^C/f^C) & : \; k^C < f^C \\ \beta & : \; k^C = f^C \\ \beta \times f^C/(k^C) & : \; k^C > f^C \end{cases}$$

The incremental factor $\triangle_{k^C,f^C}$ depends on the values of k^C and f^C in the following way: the larger the ratio of k^C to f^C, the smaller is the value of $\triangle_{k^C,f^C}$. Thus, as the number

of times C being extensible increases, the effect of a retrieval episode on the feature weights decreases. In contrast, the smaller the ratio of k^C to f^C, the closer is $\triangle_{k^C,f^C}$ to $\beta \times n^C$. Thus, the effect is the opposite: the larger the ratio of f^C to k^C, the higher the value of $\triangle_{k^C,f^C}$ (i.e., $\triangle_{k^C,f^C}$ comes closer to $\beta \times n^C$).

Case Indexing Structure. We conceived an indexing structure that discriminates cases at two levels: at the top level, the goal discrimination network (GDN) discriminates cases by the goals they achieve. It forms collections $\{Coll\}$ of cases such that within each collection *Coll* all cases achieve the same goals. The GDN is not affected by the case index revision policy so we omit further discussion (details about the GDN can be found in (Muñoz-Avila, 1998)).

The second level indexes cases within the same collection *Coll* by their weighted features. The weight of the feature relative to a collection *Coll* is determined as follows:

$$\omega_{i,Coll} = \sum_{C \in Coll} \omega_{i,C}$$

If a feature i does not occur in a case C in *Coll* its associated weight, $\omega_{i,C}$ is set to zero. The features occurring in cases in the same collection *Coll* are grouped in intervals according to their weight relative to *Coll* such that each interval has at most a predefined number, max, of features. Within an interval, features are grouped relative to their relative frequency in the collection in so-called feature-discrimination trees (Veloso, 1994). Nodes in feature-discrimination trees contain sets of features. They have the property that the more frequent a feature occurs in a collection *Coll*, the smaller the distance between the node containing that feature and the root. Thus, if a feature occurs in all cases, it is contained in the root. Conversely, if a feature occurs in a single case, it is contained in a leaf of the tree. The root will not contain any features if none of the features is common to all cases in the collection.

During retrieval the second level is traversed keeping track of the weighted proportion of features in each case in *Coll* that occur in the current problem. The first interval visited is the one containing the features with the heavier weights (i.e., the features which have the higher relevance relative to *Coll*). Retrieval continues by visiting the next interval with heavier weights and so on. Retrieval finishes when either a case C is found such that $sim_{wg}(C, P)$ is greater than a pre-defined threshold or the interval with the lighter weights has been visited and no case was found meeting this condition.

When case indexes are revised, the second level of the hierarchy may be revised as well because the weight of a feature relative to the corresponding collection of cases might change. There are two possibilities:

- The updated weight of the feature remains within the same interval as before. In this situation no further revision is necessary.
- The updated weight of the feature is no longer within the same interval. This leads to the reorganization of one or more intervals, which in turn leads to the creation of new feature-discrimination trees for each interval that was modified.

4.2. Case Retention Policy

The outcome of the retrieved case is independent of the effort required to adapt the cases. If a case is non extensible, it does not necessarily mean that a large adaptation effort took place. The first-principles planner, for example, might only need to revise a few of the planning decisions made during replay and might need to perform little search to solved

the remaining subgoals. In such situations, retaining the found solution as a new case may increase the redundancy of the case base because clearly the existing case(s) can cope with the new problem without much effort.

Conversely, the fact that the retrieved case is extensible does not necessarily mean that the adaptation effort was low. For example, if the derivational trace in the retrieved case contains a single planning decision, the first-principles planner may have to perform a large search effort to solve the remaining subgoals. Clearly in such a situation the retrieval is non beneficial. However, since the case is extensible, the newly found solution is not stored as a new case in a case retention policy retaining non extensible cases. Thus an opportunity to increase the competence of the case base is missed.

To overcome these problems we stated a case retention policy based on the contribution of the retrieved cases to the overall adaptation effort. The effort is measured in terms of the size of the search space that was traversed to solve the problem which in turn is measured in terms of the number of planning decisions made in solving the problem. During the adaptation effort, the following two values are computed: *sizeCases* (the number of decisions that were replayed from the cases) and *sizePlan* (the total number of decisions made by the underlying first-principles planner). *sizePlan* includes not only the planning decisions that lead to a solution but also the decisions that were revised (including any decisions in which revisions to replayed decisions were made). The retrieval is considered *beneficial* if the following holds:

$$sizeCases/(sizeCases + sizePlan) \geq thr$$

where thr is a predefined threshold. If this condition does not hold we say that retrieval is *non beneficial* or *detrimental*.

Our case retention policy stores found solutions as new cases only if the retrieval is detrimental. The value of the threshold thr is a parameter of the system. For example, if it is set to 2, new cases are created only if the size of the search space explored by the first principles planner is at least as large as the size of derivational traces replayed from the cases.

4.3. Case Index Revision Improving Retain Policies

We found an interesting effect of integrating both policies into a single system: if the retrieved case is extensible but its retrieval is detrimental, it means that even though no revision was necessary on the decisions taken from the case, still a significant effort was required to solve the new problem. Clearly in such a situation, retaining the found solution as a new case is necessary to fill the gap in competence of the case based planner. On the other hand, if a case is non extensible and its retrieval detrimental, its evaluation as detrimental may be the result of decisions from the retrieved case that needed to be revised. In such a situation, adding the solution as a new case may be redundant. Ideally one could try to solve the problem again without the non extensible case to determine if the retrieval is still detrimental. This is not a feasible possibility in many situations. However, because of the case index revision policy, the more frequently a case is retrieved, the less likely it is that the case is non extensible when retrieved. That is, it becomes less likely that planning decisions replayed from the case need to be revised by the first-principles planner. This means, that over a period of time the case index revision policy makes it even less likely that the case retention policy stores redundant cases. Our experiments will confirm this conclusion.

5. EMPIRICAL EVALUATION

We implemented the two CBM policies on CAPLAN/CBC, a case-based planner implementing derivational replay on top of a partial-ordered planner, CAPLAN(Muñoz-Avila & Weberskirch, 1996). We performed two experiments. In the first experiment we wanted to observe the effects of the case index revision and case retention policies locally for single cases. In the second experiment we wanted to observe the combined effect of the two policies on the overall case-based planning process.

5.1. Problem Domains

We performed experiments with the the logistics transportation domain (Veloso, 1994) and the domain of process planning (Muñoz-Avila & Weberskirch, 1996). The logistics transportation domain was originally specified in (Veloso, 1994), In this domain, features indicate the location of packages, the means of transportation available and the relation between the locations. Goals involve relocating the packages.There are different types of locations and two types of assets: trucks and airplanes, which can be used to move packages between locations. However, the assets have restriction on their use. For example, trucks can only travel between locations in the same city and airplanes can only travel between airports. This domain has a total number of 8 types of objects and 9 operations.

The second domain is the domain of process planning for manufacturing mechanical workpieces symmetrical with respect to an axis. A planning problem in this domain is given by a geometrical description of a workpiece. The description of a workpiece is built up from geometrical primitives like cylinders, cones and toroids that describe monotonic areas of the outline, possibly augmented by surface conditions. For such a planning problem a sequence of processing operations is to be found that will machine the workpiece while considering available resources (i.e. tools, machines) and technological constraints related to the use of these resources. The process begins with the clamping of a piece of raw material on a lathe machine that rotates it at a very high speed. In most situations, the outline of the workpiece cannot be machined in one step but repeated cutting operations are necessary. The domain has a total of 33 types of objects and 27 operators.

5.2. Evaluating the Case Index Revision Policy

Our first experiment measures the performance of the case index revision and case retention policies on local cases. Retrieval was performed in two modes, dynamic and static. In the dynamic mode, the case index revision process was performed. In the static mode, the weights of the features were always set to one and they remain unchanged throughout the experiment.

Experimental Setup. The experiment consisted of 5 runs. In each run, a problem, called the pivot problem, was stated. A solution for the pivot problem was found; the solution together with the problem were used to form a case, C. All feature weights of C were set to 1 and k_C, f_C were set to 0. Then, some features of the pivot problem were randomly selected. A new goal and new features that do not occur in the pivot problem were also given. Taking the pivot problem as basis, new problems were formed by changing the fixed features, or/and by adding the new goal and the new features. Changing a fixed feature means changing the relations between the objects mentioned in the feature. For example, if a feature states that a truck is in a certain location, the changed feature will state that the truck is in another location. The problem collection met the following conditions:

1. If the weight, $w_{i,C}$, of each feature, i, in C is set to 1, then the similarity, $sim_{wg}(C, P)$, between C and each problem P in the collection is greater or equal to the 75% retrieval threshold that was set for this experiment. This means that C is retrieved for each problem in the collection when retrieval was performed in static mode.
2. The number of times that fixed features were changed in the collection is constant. For example, if a fixed feature indicates the location of a truck and another fixed feature indicates that a post office is in a certain city, the number of problems in which the truck's location is the same as the number of problems in which the post office's city is changed.
3. If n denotes the number of fixed features, then problems were ordered in such a way that within a sequence of problems, $Problem_{mn+1}, ..., Problem_{mn+n}$, the number of changes of a fixed feature is constant ($m = 0, 1, ...$). For this reason, the number of problems in the collection is a multiple of the number of selected features.

In the experiments this multiple was five. In addition, in the logistics transportation domain five features were fixed and in the domain of process planning six. Thus, the collections consisted of 25 problems in the first domain and 30 in the second one. The total number of problems involved was 125 in the logistics transportation domain and 150 in the domain of process planning.

Discussion about the Experimental Setup. Conditions 2 and 3 ensure fairness; condition 2 eliminates the possibility of bias towards any feature by avoiding that any feature occurs more frequently than others. Condition 3 ensures that the distribution of the changing features is uniform through the collection.

Results. Figures 3 and 4 summarize the results of this experiment for the domain of process planning and the logistics transportation domain respectively. Each of the five runs with the dynamic mode and the overall results for the run with the static mode are shown. The first bar of each of these runs presents the percentage of times the pivot cases were retrieved. The second bar presents the percentage of times that the pivot cases were retrieved but were non extensible.

Discussion of the Experiment Results. These experiments show that case index revision increases the reliability of the retrieval by decreasing the percentage of times the case was not extensible. This can be seen by comparing the fifth run against the static run, that is, comparing the reliability of retrieval (i.e., percentage of non extensible cases retrieved, second bar) using the dynamic mode after several retrieval episodes against the reliability of retrieval in the static mode. Notice, in addition, that in the later runs the changes in the percentages tends to decrease.

5.3. Evaluating the Case Retention Policy

We compared our case retention policy based on detrimental retrievals against previous policies in which either new cases are added every time a solution is found (the most common strategy as we will see in Section 6) or only when the case is non extensible. This experiment was performed in parallel with the experiments performed in the previous section. The beneficial threshold was set to 2 and the following items were measured:

1. Percentage of detrimental retrievals (third bar of each run shown in Figures 3 and 4).

2. Percentage of case-based problem solving episodes in which the case is non extensible but its retrieval beneficial (fourth bar).
3. Percentage of case-based problem solving episodes in which the retrieved case is extensible but its retrieval is detrimental (fifth bar).

Discussion of the Experimental Results. We observe that even in the last sequences, when the percentage of retrievals of non extensible decreases, detrimental retrieval episodes are still likely to occur. Thus, the adaptation effort is independent of the fact that the retrieved case tends to be extensible. This is particularly interesting in the logistics transportation domain where the percentage of detrimental retrievals oscillates even though the percentage of retrieval decreases in a significant way. We conclude that retaining cases based on the benefits of the retrieval is a more adequate policy than retaining non extensible cases. Two arguments can be given supporting this conclusion:

- Cases may be created which can be generated from the existing cases with little effort (see the static run, fourth bar in both figures: more than 10% and more than 20% of the new solutions would have been added to the case base even though they are redundant).
- Situations may occur in which the adaptation is large but no new cases are created because the retrieved case is extensible (see the static run, fifth bar in both figures: more than 15% and 20% of new solutions would not have been added to the case base even though the adaptation effort to create them was significant).

5.4. Observing the Relation between the CBM Policies

We observe that if a case is retrieved frequently, it is less likely that the case is non extensible (see the second bar, fifth run in both figures: approximately 5% of the retrieved cases were non extensible). Thus, a case retention policy retaining non extensible cases will result in a significant decrease in cases being retained independently of the problem solving effort. We conclude that a case index revision policy based on examining if the retrieved cases are extensible or not is incompatible with a case retention policy retaining non extensible cases.

On the other hand, new cases should be created in situations in which the retrieved cases are extensible but the retrieval is still detrimental because even without revising decisions taken from the cases, the adaptation effort is large (see the fifth bar, fifth run: more than 5% and 10% of the retrievals were detrimental). Thus, we conclude that as the likelihood of retrieving non extensible cases decreases, detrimental retrieval indicates the need for a creation of a new case.

5.5. Evaluating the Impact of the CBM Policies on overall performance

In the previous sections we evaluated the performance of each of the CBM policies separately. In this section we report on an experiment to study the effect on the performance that the combined CBM polices has on the overall CBR problem solving process.

The Experimental Setup. The experiment consisted of five runs. In each run a collection of problems was randomly generated. Each collection consisted of 2 one-goal problems, 4 two-goal problems, 8 three-goal problems, 10 four-goal problems and so on until 10 eight-goal problems. No two problems were the same within a collection or in different collections. This experimental setup allowed one to better observe the effects of the learning process. In addition, it was intended to be a fair simulation of realistic situations; by not

saturating the case base first with 1-goal problems, then with 2-goal problems and so on, problem solving episodes are likely to occur in which, say, to solve an 8-goal problem only a 2-goal case is available. This is particularly significant given our previous experiments that showed the independence between the retrieved case being extensible and its retrieval being beneficial. The same case base was updated during each run according to the CBM policies. The retrieval threshold was set to 75%. The beneficial threshold was set to 4 to 1. We performed the experiments on the domain of process planning.

Results. We measured the time for solving each problem for CAPlan/CbC when no case index revision was done and every solution obtained was stored as a new case. this is the standard retrieval procedure in Prodigy/Analogy (Veloso, 1994). We also measured the results with a retrieval procedure known as dependency-driven retrieval (Muñoz-Avila, 1998), which is particularly suitable for the domain of process planning as it uses domain-specific information about geometrical relations of the workpieces. We wanted to observe how our general CBM policies compare with this specialized procedure. Both results are depicted in Figure 5 (a); the dependency-driven retrieval is labeled "Depend.-Dr.Ret". The standard retrieval procedure, named goal-driven retrieval is labeled "Goal-Dr. Ret.". The results for CAPlan/CbC under the consideration of the CBM policies is shown in Figure 5 (b). The values of the dependency- and the goal-driven retrieval and of CAPlan correspond to the average of the five runs. For CAPlan/CbC with the CBM policies, each run is correspondly represented in one curve.

Discussion of the Experimental Results. We observe that with each run the performance of CAPlan/CbC improves. Given that at the first two runs, the impact resulting from the case index revision policy is small, we conclude that the improvement there is mainly due to standard CBR guidance. This is corroborated by the fact that the values obtained are in the same range of those in Figure 5 (a) (e.g., on average for the 8 goals, the standard retrieval procedure results in an overall time of near 130 seconds and at the second run the time for the 8 goals was just about 120 seconds). Thus, the further improvements in the next runs must be due to the combination of CBM policies. A significant increase in performance is observed between runs 2 and 3 and then the rate of increase is relatively low. This suggests that at a certain point no further efficiency gains are made. However, we found that at the last run less than 10% of the new found solutions were stored, which illustrates the effectiveness of our case retention policy combined with the case index revision policy. Overall, the improvement in the case-based problem solving process is at least 40% in the fifth run. In addition, in the fifth run, the results are comparable to those of the specialized retrieval procedure (e.g., for the 8 goals, the average was almost 80 seconds whereas for the fifth run the average for the 8 goals was approximately 90 seconds).

6. RELATED WORK

Leake and Wilson (1998) presented a framework for classifying CBM systems among several dimensions, namely, the activation timing, the integration type, the scope of changes and the type of data:

- The *activation timing* can be periodic if it takes place in a pre-set point of the CBR cycle, conditional, if it takes place when a condition is met or ad-hoc if it is defined by conditions external to the CBR system.
- The *integration type* can be on-line or off-line depending on whether it takes place during

a CBR problem solving episode or not.

- The *scope of changes* can be narrow or broad depending on whether it affects a small portion of the case base or the whole case base
- The *type of data* reflects how the collection of data is done. It can be none, if no collection is done, synchronic if the collection is done by considering the current case base or diachronic if the collection is done by considering the evolution of the case base over a period of time.

We studied several case-based planners built on top of a first principles planner and classified their CBM policies according to Leake and Wilson's framework. Prodigy/Analogy (Veloso & Carbonell, 1993) implemented derivational analogy on top of a first-principles planner for the first time. Priar (Kambhampati & Hendler, 1994) introduced cases containing hierarchical plans. SPA is a case-based planner performing single-case adaptation (Hanks & Weld, 1995) and MPA uses SPA to perform multiple-case adaptation (Francis & Ram, 1995). derSNLP+EBL (Ihrig & Kambhampati, 1997) performs case adaptation with standard replay based on a partial-order planner (SNLP). MRL uses deductive planning as inferencing mechanism (Koehler, 1995). Paris stores and reuses abstract cases (Bergmann & Wilke, 1995)

Most of the systems, namely, Prodigy/Analogy, Priar, MRL, SPA/MPA and Paris record new solutions after they are found in the CBR problem solving episodes. Thus, the activation timing is periodic because it always takes place at the same point of the CBR process and their integration type is on-line because it occurs during the course of the CBR problem solving episode. Because only a case is added, the scope is narrow. The type of data is none because no data analysis is performed; the new solution is always stored.

derSNLP+EBL's CBM policy retains the solution found as a new case if a non extensible case is retrieved. Thus, the activation timing is conditional. It is also on-line because it occurs during the course of the CBR problem solving episode. Because only a case is added, the scope is narrow. The type of data is synchronic because it considers the retrieved cases.

CAPlan/CbC's case retention policy has the same classification as derSNLP+EBL: it is conditional (i.e., retains if the retrieval is detrimental), is on-line, narrow and synchronic. CAPlan/CbCs case index revision policy is also on-line. However, its activation timing is periodic because it always takes place (how the indexes are updated depends on whether the retrieved cases are extensible or not). Its scope is narrow if the feature weight intervals do not change but the scope might be large if the feature weight intervals change as a result of the revision. Another difference is that the type of data is diachronic as it considers the trend of the retrieved cases (i.e., the factors k^C and f^C). Leake and Wilson observed that diachronic data collection is the most informative.

7. FINAL REMARKS

We presented two CBM policies for *any* general purpose case-based planner adapting cases by derivational replay. The first policy revises case indexes during case-based problem solving episodes. The case revision policy is based on a feature weighting model that updates the feature weights depending on the outcome of the retrieval. The second policy retains found solutions as new cases depending on the contribution of the retrieved cases to the adaptation effort. New cases are created only if the retrieval is detrimental.

We implemented the CBM policies on the case-based planner CAPLAN/CBC and performed empirical evaluations. We concluded that the case index revision policy improves the accuracy of the retrieval. We also compared the case retention policy with policies in which

either new cases are added every time a solution is found or only when the retrieved cases are not extensible. Our results indicate that our retention policy is a better filter for redundant cases. We observed that by fine tuning the feature weights, the case index revision policy improves the filtering process of the case retention policy.

References

Althoff, K.-D., Auriol, E., Barletta, R., and Manago, M. (1995). *A Review of Industrial Case-Based Reasoning Tools.* AI Perspective Report, Oxford, UK: AI Intelligence.

Barrett, A. and Weld, D. (1994). Partial-order planning: Evaluating possible efficiency gains. *Artificial Intelligence*, 67(1):71–112.

Bergmann, R. and Wilke, W. (1995). Building and refining abstract planning cases by change of representation language. *Journal of Artificial Intelligence Research*, 3:53–118.

Francis, A. G. J. and Ram, A. (1995). A domain-independent algorithm for multi-plan adaptation and merging in least-commitment planning. *AAAI Fall Symposium: Adaptation of Knowledge Reuse*, Menlo Park, CA. AAAI Press.

Hanks, S. and Weld, D. (1995). A domain-independent algorithm for plan adaptation. *Journal of Artificial Intelligence Research*, 2.

Ihrig, L. & Kambhampati, S. (1997). Storing and Indexing plan derivations through explanation-based analysis of retrieval failures. *Journal of Artificial Intelligence.* Vol 7, pp. 161-198.

Kambhampati, S. & Hendler, J. (1994). Exploiting causal structure to control retrieval and refitting during plan reuse. *Computational Intelligence*, 10(2):213–244.

Koehler, J. (1994). Flexible plan reuse in a formal framework. *Current Trends in AI Planning*, IOS Press.

Koehler, J. & Shuster, B. (2000). Elevator Control as a Planning Problem. *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2000).* Breckenridge, CO: AAAI Press.

Leake, D.B., & Wilson, D.C. (1998). Categorizing case-base maintenance: dimensions and directions. *Proceedings of the Fourth European Workshop on Case-Based reasoning (EWCBR-98).* Dublin, Ireland: Springer.

Lei, C., Babka, O., Garanito, L.A.G. (1999). Case Library Reduction Applied to Pile Foundation. *Proceedings of the Third International Conference on Case-Based Reasoning (ICCBR-99).* Munich, Germany: Springer.

Markovitch, S., & Scott, P.D. (1993). Information Filtering: selection Mechanisms in learning systems. *Machine Learning*, 10.

Minton, S. (1990). Quantitative results concerning the utility of explanation-based learning. *Artificial Intelligence*, 42:363–391.

Muñoz-Avila, H., Paulokat, J., and Wess, S. (1994). Controlling non-linear hierarchical planning by case replay. *Procedings of the Second European Workshop on Case-Based Reasoning (EWCBR-94).* Paris, France: Springer.

Muñoz-Avila, H. & Weberskirch, F. (1996). Planning for manufacturing workpieces by storing, indexing and replaying planning decisions. *Proceedings of the 3nd International Conference on AI Planning Systems (AIPS-96).* Edinburgh, UK: AAAI-Press.

Muñoz-Avila, H. (1998). Integrating Twofold Case Retrieval and Complete Decision Replay in CAPlan/CbC. *PhD Thesis*, University of Kaiserslautern.

Muñoz-Avila, H. (1999). A case retention policy based on detrimental retrieval. *Proceedings of the Third International Conference on Case-Based Reasoning (ICCBR-99).* Munich, Germany: Springer.

Nebel, B. & Koehler, J. (1995). Plan Reuse versus Plan Generation: A Theoretical and

Empirical Analysis, *Artificial Intelligence*, 76(1-2): 427-454, 1995.

Racine, K., & Yang, Q. (1997) Maintaining unstructured case bases. *Proceedings of the Second International Conference on Case-Based Reasoning (ICCBR-97).* Providence: Springer.

Salzberg, S. L. (1991). A nearest hyperrectangle learning method. *Machine Learning*, 1.

Smyth, B. and Keane, M. (1994). Retrieving adaptable cases. *Proceedings of the Second European Workshop on Case-Based Reasoning (EWCBR-94).* Paris, France: Springer.

Smyth, B., & Keane, M.T., (1995). Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. *Proceedings of the International Joint Conference on Artificial Intelligence.* Montreal: Morgan Kaufmann.

Smyth, B., & McKenna, E. (1999). Retrieving adaptable cases. *Proceedings of the Third International Conference on Case-Based Reasoning (ICCBR-99).* Munich, Germany: Springer.

Veloso, M. & Carbonell, J. (1993). Derivational analogy in prodigy: Automating case acquisition, storage, and utilization. *Machine Learning*, 10.

Veloso, M. (1994). *Planning and learning by analogical reasoning.* Number 886 in Lecture Notes in Artificial Intelligence. Springer.

FIGURE CAPTIONS

1. Figure 1: (a) a planning decision made by a first-principles planner and (b) a planning decision made by derivational replay.
2. Figure 2: Example of a non extensible case.
3. Figure 3: Results for the Domain of process planning.
4. Figure 4: Results for the logistics transportation domain.
5. Figure 5: Performance gains with (a) standard retrieval and (b) the complete case-based planner.

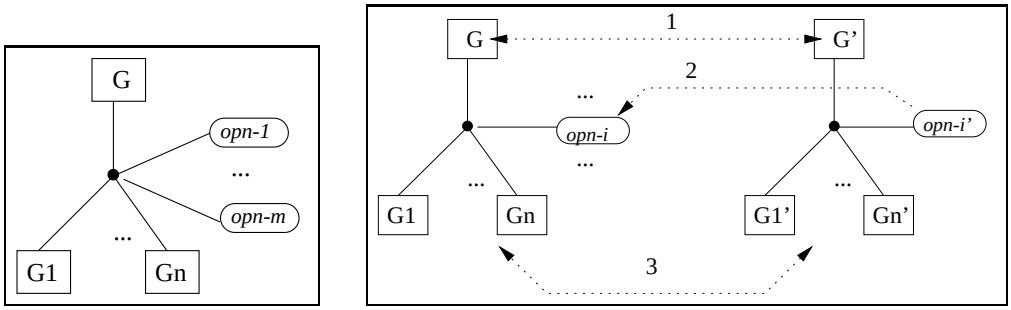

Figure 1, Héctor Muñoz-Avila, Case-Base Maintenance by Integrating Case Index Revision and Case Retention Policies in a Derivational Replay Framework

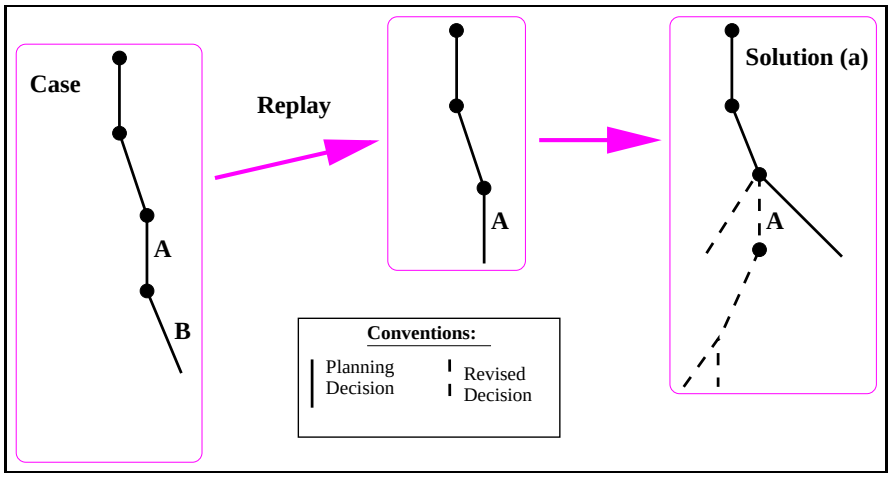

Figure 2, Héctor Muñoz-Avila, Case-Base Maintenance by Integrating Case Index Revision and Case Retention Policies in a Derivational Replay Framework

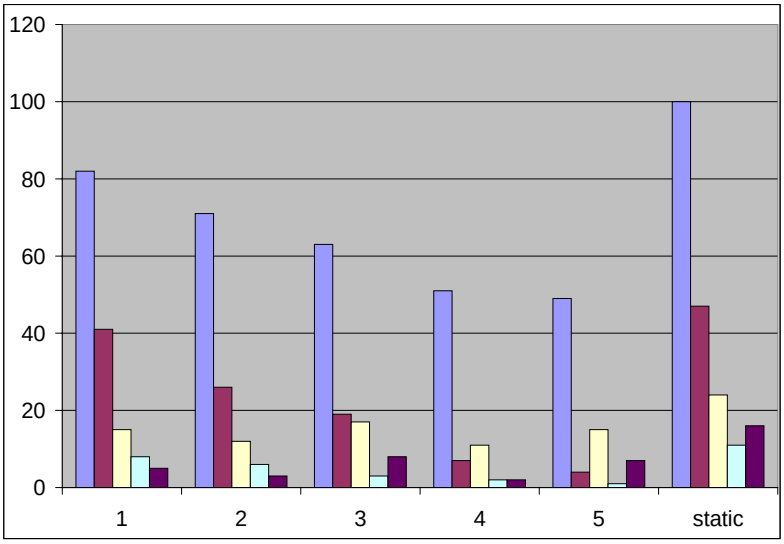

Figure 3, Héctor Muñoz-Avila, Case-Base Maintenance by Integrating Case Index Revision and Case Retention Policies in a Derivational Replay Framework

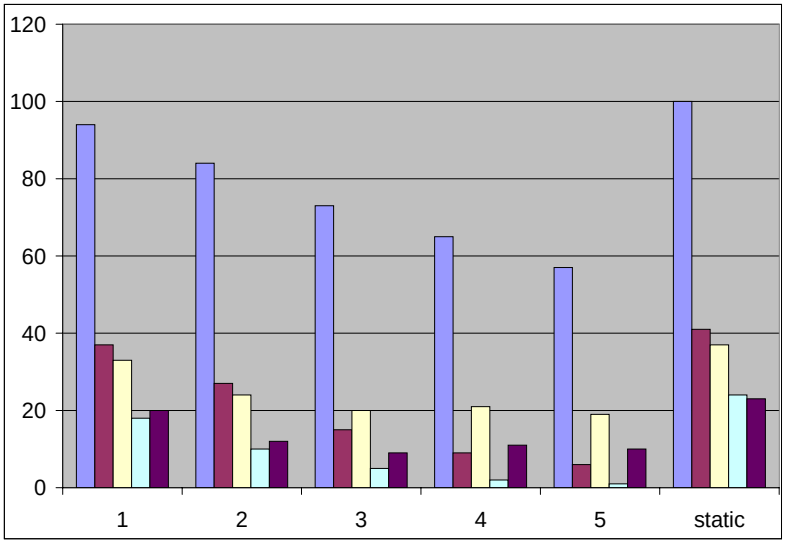

Figure 4, Héctor Muñoz-Avila, Case-Base Maintenance by Integrating Case Index Revision and Case Retention Policies in a Derivational Replay Framework

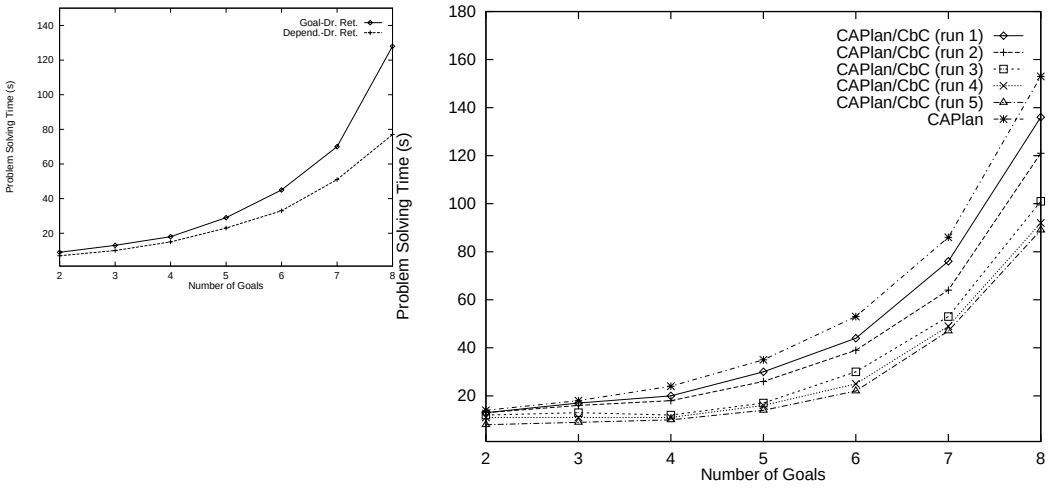

Figure 5, Héctor Muñoz-Avila, Case-Base Maintenance by Integrating Case Index Revision and Case Retention Policies in a Derivational Replay Framework